What is Machine Learning?

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

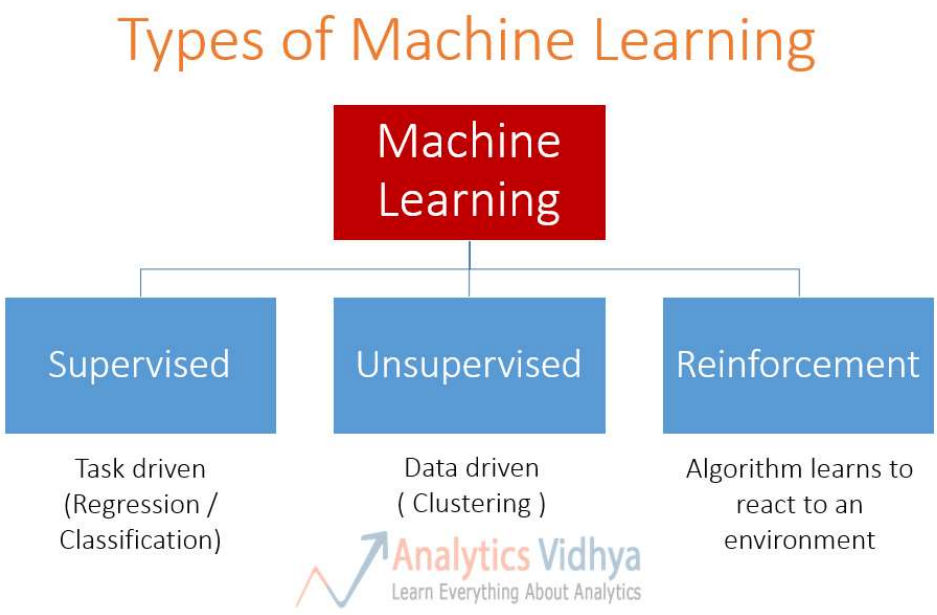
For example, a computer program that learns to play checkers might improve its performance as measured by its ability to win at the class of tasks involving playing checkers games, through experience obtained by playing games against itself. In general, to have a well-defined learning problem, we must identity these three features: the class of tasks, the measure of performance to be improved, and the source of experience.

A checkers learning problem:

- Task T: playing checkers
- Performance measure P: percent of games won against opponents
- Training experience E: playing practice games against itself

We can specify many learning problems in this fashion, such as learning to recognize handwritten words, or learning to drive a robotic automobile autonomously.

What are the types of Machine Learning algorithms?



Supervised Learning / Predictive models:

Predictive model as the name suggests is used to predict the future outcome based on the historical data. Predictive models are normally given clear instructions right from the

beginning as in what needs to be learnt and how it needs to be learnt. These class of learning algorithms are termed as **Supervised Learning.**

For example: Supervised Learning is used when a marketing company is trying to find out which customers are likely to churn. We can also use it to predict the likelihood of occurrence of perils like earthquakes, tornadoes etc. with an aim to determine the Total Insurance Value. Some examples of algorithms used are: Nearest neighbour, Naïve Bayes, Decision Trees, Regression etc.

Unsupervised learning / Descriptive models:

It is used to train descriptive models where no target is set and no single feature is important than the other. The case of unsupervised learning can be: When a retailer wishes to find out what are the combination of products, customers tends to buy more frequently. Furthermore, in pharmaceutical industry, unsupervised learning may be used to predict which diseases are likely to occur along with diabetes. Example of algorithm used here is: K-means Clustering Algorithm

Reinforcement learning (RL):

It is an example of machine learning where the machine is trained to take specific decisions based on the business requirement with the sole motto to maximize efficiency (performance). The idea involved in reinforcement learning is: The machine/ software agent trains itself on a continual basis based on the environment it is exposed to, and applies it's enriched knowledge to solve business problems. This continual learning process ensures less involvement of human expertise which in turn saves a lot of time!

An example of algorithm used in RL is Markov Decision Process.

A good example to understand the difference is self driving cars. Self driving cars use Reinforcement learning to make decisions continuously – which route to take? what speed to drive on? are some of the questions which are decided after interacting with the environment. A simple manifestation for supervised learning would be to predict fare from a cab going from one place to another.

How to choose the right algorithm ?

With all the different algorithms in table 1.2, how can you choose which one to use? First, you need to consider your goal. What are you trying to get out of this? (Do you want a probability that it might rain tomorrow, or do you want to find groups of voters with similar interests?) What data do you have or can you collect? Those are the big questions. Let's talk about your goal.

If you're trying to predict or forecast a target value, then you need to look into supervised learning. If not, then unsupervised learning is the place you want to be. If you've chosen supervised learning, what's your target value? Is it a discrete value like Yes/No, 1/2/3, A/B/C, or Red/Yellow/Black? If so, then you want to look into classification. If the target value can take on a number of values, say any value from 0.00 to 100.00, or -999 to 999, or +to -, then you need to look into regression.

If you're not trying to predict a target value, then you need to look into unsupervised learning. Are you trying to fit your data into some discrete groups? If so and that's all you need, you should look into clustering. Do you need to have some numerical estimate of how strong the fit is into each group? If you answer yes, then you probably should look into a density estimation algorithm.

The rules I've given here should point you in the right direction but are not unbreakable laws. In chapter 9 I'll show you how you can use classification techniques for regression, blurring the distinction I made within supervised learning. The second thing you need to consider is your data. You should spend some time getting to know your data, and the more you know about it, the better you'll be able to build a successful application. Things to know about your data are these: Are the features nominal or continuous? Are there missing values in the features? If there are missing values, why are there missing values? Are there outliers in the data? Are you looking for a needle in a haystack, something that happens very infrequently? All of these features about your data can help you narrow the algorithm selection process.

With the algorithm narrowed, there's no single answer to what the best algorithm is or what will give you the best results. You're going to have to try different algorithms and see how they perform. There are other machine learning techniques that you can use to improve the performance of a machine learning algorithm. The relative performance of two algorithms may change after you process the input data. We'll discuss these in more detail later, but the point is that finding the best algorithm is an iterative process of trial and error.

Many of the algorithms are different, but there are some common steps you need to take with all of these algorithms when building a machine learning application. I'll explain these steps in the next section.

Steps in developing a machine learning application

1. **Collect data**: You could collect the samples by scraping a website and extracting data, or you could get information from an RSS feed or an API. You could have a device collect wind speed measurements and send them to you, or blood glucose levels, or anything you can measure. The number of options is endless. To save some time and effort, you could use publicly available data.
2. **Prepare the input data**: Once you have this data, you need to make sure it's in a useable format. The format we'll be using in this book is the Python list. We'll talk about Python more in a little bit, and lists are reviewed in appendix A. The benefit of having this standard format is that you can mix and match algorithms and data sources.

   You may need to do some algorithm-specific formatting here. Some algorithms need features in a special format, some algorithms can deal with target variables and features as strings, and some need them to be integers. We'll get to this later, but the algorithm-specific formatting is usually trivial compared to collecting data.
3. **Analyze the input data**: This is looking at the data from the previous task. This could be as simple as looking at the data you've parsed in a text editor to make sure steps 1 and 2 are actually working and you don't have a bunch of empty values. You can also look at the data to see if you can recognize any patterns or if there's anything obvious, such as a few data points that are vastly different from the rest of the set. Plotting data in one, two, or three dimensions can also help. But most of the time you'll have more than three features, and you can't easily plot the data across all features at one time. You could, however, use some advanced methods we'll talk about later to distill multiple dimensions down to two or three so you can visualize the data.
4. If you're working with a production system and you know what the data should look like, or you trust its source, you can skip this step. This step takes human involvement, and for an automated system you don't want human involvement. The value of this step is that it makes you understand you don't have garbage coming in.
5. **Train the algorithm:** This is where the machine learning takes place. This step and the next step are where the "core" algorithms lie, depending on the algorithm. You feed the algorithm good clean data from the first two steps and extract knowledge or information. This knowledge you often store in a format that's readily useable by a machine for the next two steps.

   In the case of unsupervised learning, there's no training step because you don't have a target value. Everything is used in the next step.

6. **Test the algorithm:** This is where the information learned in the previous step is put to use. When you're evaluating an algorithm, you'll test it to see how well it does. In the case of supervised learning, you have some known values you can use to evaluate the algorithm. In unsupervised learning, you may have to use some other metrics to evaluate the success. In either case, if you're not satisfied, you can go back to step 4, change some things, and try testing again. Often the collection or preparation of the data may have been the problem, and you'll have to go back to step 1.
7. **Use it**: Here you make a real program to do some task, and once again you see if all the previous steps worked as you expected. You might encounter some new data and have to revisit steps 1–5.

Issues in Machine Learning

Our checkers example raises a number of generic questions about machine learning. The field of machine learning, and much of this book, is concerned with answering questions such as the following:

- What algorithms exist for learning general target functions from specific training examples? In what settings will particular algorithms converge to the desired function, given sufficient training data? Which algorithms perform best for which types of problems and representations?
- How much training data is sufficient? What general bounds can be found to relate the confidence in learned hypotheses to the amount of training experience and the character of the learner's hypothesis space?
- When and how can prior knowledge held by the learner guide the process of generalizing from examples? Can prior knowledge be helpful even when it is only approximately correct?
- What is the best strategy for choosing a useful next training experience, and how does the choice of this strategy alter the complexity of the learning problem?
- What is the best way to reduce the learning task to one or more function approximation problems? Put another way, what specific functions should the system attempt to learn? Can this process itself be automated?
- How can the learner automatically alter its representation to improve its ability to represent and learn the target function?

What are the applications of Machine Learning?

It is very interesting to know the applications of machine learning. Google and Facebook use ML extensively to push their respective ads to the relevant users. Here are a few applications that you should know:

Banking & Financial services: ML can be used to predict the customers who are likely to default from paying loans or credit card bills. This is of paramount importance as machine learning would help the banks to identify the customers who can be granted loans and credit cards.

Healthcare: It is used to diagnose deadly diseases (e.g. cancer) based on the symptoms of patients and tallying them with the past data of similar kind of patients.

Retail: It is used to identify products which sell more frequently (fast moving) and the slow moving products which help the retailers to decide what kind of products to introduce or remove from the shelf. Also, machine learning algorithms can be used to find which two / three or more products sell together. This is done to design customer loyalty initiatives which in turn help the retailers to develop and maintain loyal customers.

These examples are just the tip of the iceberg. Machine learning has extensive applications practically in every domain.