Applications and Development of Machine Learning

TARIK ONALAN

Interlake High School

In the last decade, machine learning has taken centre stage in the computing field. With the advent of increasingly powerful computers, giga-, tera-, and even peta-scale computations have become possible. In this paper, we will investigate the applications of machine learning in these large-scale computations—big data—and how they can affect future development.

1. INTRODUCTION

The term “Machine Learning” was coined in 1959 by Arthur Samuel, in his paper *Some studies in machine learning using the game of checkers*, described as “a field of study that gives computers the ability to learn without being explicitly programmed”. The definition still holds, but machine learning, as a science, is now leaps and bounds ahead of where it was in 1959. Computers are now capable of analyzing giga- and tera-scale sets of data in hours or minutes instead of days, weeks, or even months or years.

It is essentially impossible to understand big data without machine learning. The computational power required to parse and analyze modern data-sets is well beyond the power of all human minds ever to be created. With computers, however, we can run calculations faster and with higher precision than any human ever could.

The applications of machine learning are vast, spanning from finance, to health care, to social media. Anywhere where there is a large pool of data to be analyzed, machine learning becomes a viable option.

2. LEARNING METHODS

2.1. Supervised Learning

Supervised learning is achieved by training a model by optimizing it to fit a set of input-output pairs. Given a set of inputs and outputs $\{(x_1, y_1), ..., (x_n, y_n)\}$, the model that is being trained will seek a function $h : X \to Y$, where X is the input and Y is the output. The key fact to remember with this learning method is that there is an expected output for every input against which the model's predictions can be compared and tuned.

A good way of visualizing this kind of learning algorithm is with a pond and a fisherman. Let us imagine a pond with 3 species of fish: catfish, carp, and goldfish. Each species swims in a particular region of the pond. The fisherman does not know which part of the pond each species prefers, however.

The fisherman takes a fish from the pond, and notes its type. They continue to do this, noting the fish types. They now know that if they take a fish from (for example) the left of the pond, it is most likely going to be a goldfish. They also know that fish from the bottom of the pond are catfish, and fish from the right of the pond are most likely carp.

Later, someone asks the fisherman to identify a fish from the pond, except they only tell the fisherman what part of the pond the fish was from—the bottom, in this case. The fisherman, having learned that fish from the bottom of the pond are most likely catfish, replies that the fish is most likely a catfish.

2.2. Unsupervised Learning

Unsupervised learning is, as the name suggests, the opposite of supervised learning. Instead of constructing a model from input-output pairs, unsupervised learning consists of constructing a model solely from an input set. Essentially, given unlabeled data, the objective in unsupervised learning is to find any hidden structures that may be present in the data, and use those hidden structures to construct a classification or regression model. Unsupervised learning does not have to be used alone: it can also be a means to an end; after understanding the structure of a dataset, it becomes easier to transition to more robust learning methods like supervised learning (2.1) or reinforcement learning (2.2)

This algorithm is slightly more difficult to visualize. Let us continue with our metaphor with the pond and the fisherman; this time, however, the fisherman, after pulling a fish out of the water, is not able to tell exactly what kind of fish it is. All they can know is that it came from, say, the left of the pond.

This time, when someone asks the fisherman to identify a fish from the pond, all the fisherman can answer, given their information, is that the fish is of the type that comes from, for example, the left of the pond. Essentially, the fisherman grouped the fish into different categories: the fish from the left are of one type, the fish from the bottom of are another type, and the fish from the right are of yet another type.

2.3. Reinforcement Learning

Reinforcement learning functions in the following manner: model is presented with an input space, and a function to calculate reward; however, in this case, there are no explicit input-output pairs, and if the model begins to stray to a sub-optimal path, there will be no outright correction of the digression. This method of learning is preferred for on-line (real-time) learning, as a constant dataset is not required for reinforcement learning. Instead, there must be a balance between exploration of new knowledge and exploitation of old knowledge.

Visualizing this type of learning is slightly different than the previous two. We can use the classic multi-armed bandit problem to describe reinforcement learning ([1]). A bandit must decide which one of k non-identical slot machines to play in order to maximize their reward. If the bandit is on a well-paying slot machine, the incentive for them to move—exploring new knowledge—is less, while their incentive to keep playing the slot machine—exploiting old knowledge—is higher. The opposite is true if the bandit is on a poorly-paying slot machine; the bandit is more likely to switch machines to try to find a better-paying machine.

3. ALGORITHMS

3.1. k-NN

k-nearest neighbors, commonly shortened as k-NN, is a popular machine learning algorithm because of its relative simplicity to implement. It is non-parametric—meaning it does not assume that the input data has any particular structure, making it an ideal tool for "testing" the structure of a dataset and identifying any basic patterns. This property also makes it perform well in unsupervised learning (2.2).

When classifying an input x_n, for example, it takes the "majority vote" of the k nearest neighbors to the input x_n to determine the class of the input.

$$x_\theta = \underset{\theta}{\operatorname{argmax}} |k_\theta \in \{k^*\}|| \tag{3.1.1}$$

Equation (3.1.1) demonstrates the "majority vote" method. The class of an input vector x, represented here by x_θ is defined as the class—θ—for which the number of nearest neighbors k_θ in in all of the possible nearest neighbors $\{k^*\}$ is maximized.

Variations of the algorithm also vary the "weight" of the vote with the inverse of the distance. This helps counteract any bias incurred by skewed data; for example, if a point were to have three neighbors, one very near and two very far, the "weighted vote" would ideally have the nearer neighbor have more influence over the class of the input vector than the further neighbors.

$$x_\theta = \underset{\theta}{\operatorname{argmin}} \frac{\sum_n \operatorname{dist}^m(k_n \in \{k_\theta^*\})}{|k_\theta \in \{k^*\}||} \tag{3.1.2}$$

Equation (3.1.2) demonstrates the "weighted vote" method. The class of an input vector x, again represented by x_θ, is defined by the θ for which the sums of the distances between the k nearest neighbors of class θ divided by the cardinality of the set $k_\theta \in \{k^*\}$ is minimized. Important to note is that the function dist is taken to the power of m; the weighting need not be inverse-linear: it could be inverse-square—arguably the most popular—inverse-cubic, inverse-quartic, et cetera.

3.2. Support Vector Machines

Support vector machines are supervised learning models (2.1) that are commonly used for classification and regression tasks. With a linear kernel (which is all we will go over, for the sake of simplicity), a support vector machine will fit a linear model to separate a dataset of two classes (while more than two can be classified, it requires extra computation).

A linear model for a support vector machine is as follows:

$$\omega \cdot x_i + b = y_i \tag{3.2.1}$$

where ω is the matrix of weights, x_i is the input vector, b is the bias, and y_i is the class. The support vector machine finds the parameters ω and b for the hyperplane that separate the input dataset into two classes: $\{-1, 1\}$. This is accomplished by maximizing the distance to elements of either class on both sides of the hyperplane. Maximizing the distance is accomplished with the aid of two solutions to the hyperplane defined in equation (3.2.1):

$$\begin{cases} \omega \cdot x_i + b \geq 1 & \text{Class above line} \\ \omega \cdot x_i + b \leq -1 & \text{Class below line} \end{cases} \tag{3.2.2}$$

As y_i is always either -1 or 1, we can simplify this further:

$$y_i(\omega \cdot x_i + b) \geq 1 \quad \forall i \in [1, n] \tag{3.2.3}$$

Finally we can take the equations in (3.2.2) to derive our objective function:

$$\begin{cases} \omega \cdot x_i + b = 1 \\ \omega \cdot x_j + b = -1 \end{cases}$$

What we are assuming here is that x_i represents the element of class 1 nearest to our line, and that x_j represents the element of class -1 nearest to our line.

$$\omega \cdot (x_i - x_j) = 2$$

$$(x_i - x_j) = \frac{2}{\|\omega\|} \tag{3.2.4}$$

Now, if we remember, we were trying to maximize the difference between the elements of the respective classes $\{-1, 1\}$ that were closest to our line; we can use equation (3.2.4) to define our objective function with respect to ω:

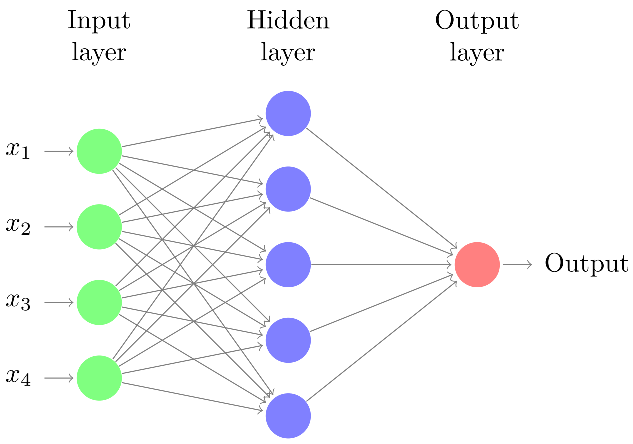
$$\underset{(i,j)}{\operatorname{argmax}}\, x_i - x_j$$

$$\underset{\omega}{\operatorname{argmin}}\, \frac{2}{\|\omega\|} \tag{3.2.5}$$

With the objective function, we can now solve for ω by putting equation (3.2.5) under the constraint of equation (3.2.3), and solving for the weights.

3.3. Neural Network

Neural networks are arguably the most publicized of the machine learning algorithms, likely due to their functionality. In essence, they function the same way a small brain would, weighing inputs and providing an output based on a threshold function. Each perceptron—the neural network's equivalent of a neuron—weighs one metric, and passes on the value to the next perceptron.



Neural networks are rather popular in practical use as well, mostly because they are able to relatively accurately map unknown functions of multiple parameters. Additionally, they are capable of learning with all three learning methods shown in section (2). This makes them a versatile tool in any data scientist's toolbox.

Let us do the derivation for a simple single hidden layer perceptron network with backpropagation. The first step is to determine an activation function; we will choose the sigmoid function:

$$\varphi(x) = \frac{1}{1 + e^x} \tag{3.3.1}$$

Now that we have an activation function, we can define the system of equations describing a neural network with one hidden layer:

$$\begin{cases} S_i(x) = \sum_n \varphi(\omega_{ni} x_n) & \text{Input layer} \\ S_j(x) = \sum_n \varphi(\omega_{nj} S_i(x)) & \text{Hidden layer} \\ S_k(x) = \omega_k S_j(x) & \text{Output Layer (bias)} \end{cases} \tag{3.3.2}$$

From there, we can describe our error function:

$$E(x_i^*) = \sum_{x_i \in x_i^*} (S_k(x_i) - y_i)^2 \tag{3.3.3}$$

The error function described in equation (3.3.3) can then be used to update the individual weights

$$\begin{cases} \omega_i = \omega_i - \alpha \frac{\partial E}{\partial \omega_i} \\ \omega_j = \omega_j - \alpha \frac{\partial E}{\partial \omega_j} \\ \omega_k = \omega_k - \alpha \frac{\partial E}{\partial \omega_k} \end{cases} \tag{3.3.4}$$

where $\alpha \in (0, 1]$, usually between 0.02 and 0.05, is used to control the speed of the gradient descent.

4. FINANCE

Machine learning has become heavily relied upon in the field of finance, partially due to the fact that the regression of market data is one of many variables—other stock prices, financial outlooks, financial news—and is very "noisy", with many random variations in a stock price in a given day. As such, neural networks—are heavily used in the field of finance, as they are exceedingly good at mapping unknown functions of multiple variables (3.3). A neural network could predict the rise and fall of certain stocks given a large feature vector containing, say, the profit margin of a company, the national inflation rate, the number of jobs created the previous quarter, et cetera. At the right α values (3.3), a neural network could also be very resistant to "overfitting": describing statistical "noise" more than the actual data.

Especially now, with the production of increasingly powerful supercomputers, more and more people are using machine learning in financial applications. Supercomputers can be used to not only calculate the trends of stocks, but calculate them in real time, learning on-line. Advanced algorithms take into account not only current stocks, but global market outlook, important corporate or governmental incidents, and similar qualitative data in addition to the raw quantitative data to augment their estimates of possible future trends. This proves itself to be a more robust method of regression, because on its own, financial data is quite difficult to regress; a dataset of stock prices from six months ago to now would not guarantee a valid model of the future. However, with more qualitative data, it becomes possible to predict market prices with slightly higher accuracy.

Important to note, of course, is that the algorithms used in financial applications are often specifically tailored for those applications. Herbert Simon, a computer scientist, economist, and psychologist, in his 1976 paper introduced a more human-like, symbol-based approach to modeling financial markets [9]. This approach was very much literal; much like what a "symbol" is to a human, the computer would analyze a situation, compare it to previously encountered situations, and predict a result based on what the previous examples were. However, there were inherent difficulties in enacting the approach: where did the "symbols" come from? How do they adapt to new situations? How are they influenced by their environment? In recent history, there seems to have been a partial resolution to these questions in the form of autonomous agents: not quite symbol-based, but making its own decisions based on past data, buying and selling stocks [3, 2].

5. HEALTH CARE

Machine learning in the health care sector is centred more around clustering; support vector machines (3.2), decision trees, k-nearest neighbor (3.1), and k-means clustering are some of the more popular algorithms in use in the computational biology fields, specifically. With the advent of genome research, scientists have to search for genetic anomalies that could be signs of future disease. Clustering algorithms like k-NN (3.1) are useful in this application because the computational power required is not as demanding as an unsupervised neural network, while they are still able to identify any important structures in the data. These unsupervised clustering algorithms are generally paired with an overarching supervised algorithm; after identifying clusters of possible genes, the supervised algorithm takes over and begins mapping the clusters to symptoms or diseases.

Genomes themselves pose an interesting computational challenge; a single genome is around 1.5 gigabytes, which means that loading and processing a genome is very memory and computation intensive. After loading a genome, the next objective is identifying outliers. However, every human can have different mutations on different genes, some of which may not have any significant health-related effects. Genomes must be cross-checked with wide varieties of populations to train an algorithm to ignore certain genes and be suspicious of others.

Let us also note that machine learning can be used in other regions of health care as well. Just as the ability to effectively generalize data and ignore "noise" makes machine learning effective in financial applications, it also makes machine learning effective in health care. EEG, ECG, heart rate, blood pressure, and similar measurements can all be processed at once using machine learning, while reducing the likelihood that noise would significantly affect the prognosis.

Arguably the most important part of machine learning in health care, however, is that it helps save human lives: take, for example lymphoma. Diffuse large B-Cell lymphoma, the most common lymphoid malignancy in adults, is curable in less than 50% of cases. However, with machine learning, doctors are able to identify the type of chemotherapy drug a tumor is most susceptible to, possibly increasing the survival rate of the patient ([14]).

6. SOCIAL MEDIA

Machine learning in social media is a trend that has developed fairly recently. The common algorithms in use in social media—for the purpose of news displays, post displays, et cetera—are quite varied, ranging from classification, to clustering, to regression. This is because the possible applications of machine learning in social media are varied and numerous: advertising, recommendations, profiling, predictions, et cetera.

Advertising and recommendations are (often to our chagrin) becoming increasingly common on the internet. They have become a lucrative source of revenue for many companies, some of which have access to millions of gigabytes of data. User data in particular is what is important; every advertisement is tailored towards a certain user. Machine learning algorithms analyze everything from a user's search history, website interactions, comments, posts, et cetera, to decide which advertisement or recommendation to place in the user's browser window ([11]).

Machine learning can also be used to predict characteristics about a user: male or female, liberal or conservative, introvert or extrovert ([10]). While this does inevitably tie back to advertising and recommendations, it presents an slightly different problem. Sentences need to be encoded into a form machine learning models can parse; natural language processing is required to analyze posts and comments for biases; analysis of known users is required for a "baseline". Essentially, while user information is used in choosing what advertisement goes where, extracting that information is another problem entirely.

In addition, the rise of what is now known as the "personal assistant" on many electronic devices requires the prediction of user actions to streamline user experience. By training a model on a user's previous actions, that model can then be used to predict that user's future actions ([15]). It is important to note that we believe the prediction of user actions still falls within the realm of social media, as it is directly dealing with the end user and their actions.

7. CONCLUSION

Machine learning is a field with a list of applications that is constantly growing in our age of big data and instant information. Every day, 2.5 exabytes (2.5 billion gigabytes) of data are created on the internet, more than any human—or computer, for that matter—can handle in our day and age.

Simply speaking, however, humans are no longer—and never quite were—a viable option for analyzing large, multidimensional datasets; they are simply too slow and too inaccurate. A computer paired with the proper algorithm, however, can be accurate and performant. It is fascinating to think what could be extrapolated from just one percent of the internet's daily data production.

Machine learning will inevitably continue to grow in prominence in our daily lives, becoming an increasingly integral part of our society as the amount of data we generate grows. This, essentially, brings us back to our main point: it is impossible to understand big data without machine learning.

REFERENCES

[1] P. Auer et al. *Gambling in a rigged casino: The adversarial multi-armed bandit problem.* Oct. 1995, pp. 322–331. DOI: 10.1109/SFCS.1995.492488.

[2] R.C. Cavalcante and A.L.I. Oliveira. "An autonomous trader agent for the stock market based on online sequential extreme learning machine ensemble". In: *Neural Networks (IJCNN), 2014 International Joint Conference on.* July 2014, pp. 1424–1431. DOI: 10.1109/IJCNN.2014.6889870.

[3] Shu-Heng Chen. "Computational intelligence in economics and finance: Carrying on the legacy of Herbert Simon". In: *Information Sciences* 170.1 (2005). Computational Intelligence in Economics and Finance, pp. 121–131. ISSN: 0020-0255. DOI: http://dx.doi.org/10.1016/j.ins.2003.11.006. URL: http://www.sciencedirect.com/science/article/pii/S0020025503004444.

[4] Luc Devroye et al. "On the Strong Universal Consistency of Nearest Neighbor Regression Function Estimates". English. In: *The Annals of Statistics* 22.3 (1994), pp. 1371–1385. ISSN: 00905364. URL: http://www.jstor.org/stable/2242230.

[5] Clive W.J. Granger. "Forecasting stock market prices: Lessons for forecasters". In: *International Journal of Forecasting* 8.1 (1992), pp. 3–13. ISSN: 0169-2070. DOI: http://dx.doi.org/10.1016/0169-2070(92)90003-R. URL: http://www.sciencedirect.com/science/article/pii/016920709290003R.

[6] Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola. "Kernel Methods in Machine Learning". English. In: *The Annals of Statistics* 36.3 (2008), pp. 1171–1220. ISSN: 00905364. URL: http://www.jstor.org/stable/25464664.

[7] Matjaž Kukar et al. "Analysing and improving the diagnosis of ischaemic heart disease with machine learning". In: *Artificial Intelligence in Medicine* 16.1 (1999). Data Mining Techniques and Applications in Medicine, pp. 25–50. ISSN: 0933-3657. DOI: http://dx.doi.org/10.1016/S0933-3657(98)00063-3. URL: http://www.sciencedirect.com/science/article/pii/S0933365798000633.

[8] Javier M. Moguerza and Alberto Muñoz. "Support Vector Machines with Applications". English. In: *Statistical Science* 21.3 (2006), pp. 322–336. ISSN: 08834237. URL: http://www.jstor.org/stable/27645765.

[9] Allen Newell and Herbert A. Simon. "Computer Science As Empirical Inquiry: Symbols and Search". In: *Commun. ACM* 19.3 (Mar. 1976), pp. 113–126. ISSN: 0001-0782. DOI: 10.1145/360018.360022. URL: http://doi.acm.org/10.1145/360018.360022.

[10] Marco Pennacchiotti and Ana-Maria Popescu. "A Machine Learning Approach to Twitter User Classification". In: 2011. URL: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/view/2886.

[11] C. Perlich et al. "Machine learning for targeted display advertising: transfer learning in action". English. In: *Machine Learning* 95.1 (2014), pp. 103–127. ISSN: 0885-6125. DOI: 10.1007/s10994-013-5375-2. URL: http://dx.doi.org/10.1007/s10994-013-5375-2.

[12] Paul Sajda. "MACHINE LEARNING FOR DETECTION AND DIAGNOSIS OF DISEASE". In: *Annual Review of Biomedical Engineering* 8.1 (2006). PMID: 16834566, pp. 537–565. DOI: 10.1146/annurev.bioeng.8.061505.095802. eprint: http://dx.doi.org/10.1146/annurev.bioeng.8.061505.095802. URL: http://dx.doi.org/10.1146/annurev.bioeng.8.061505.095802.

[13] Arthur L. Samuel. "Some studies in machine learning using the game of Checkers". In: *IBM JOURNAL OF RESEARCH AND DEVELOPMENT* (1959), pp. 71–105.

[14] Margaret A. Shipp et al. "Diffuse large B-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning". In: *Nat Med* 8.1 (Jan. 2002), pp. 68–74. ISSN: 1078-8956. DOI: 10.1038/nm0102-68. URL: http://dx.doi.org/10.1038/nm0102-68.

[15] Leonard Cleve Stuart. "User Modeling via Machine Learning and Rule-Based Reasoning to Understand and Predict Errors in Survey Systems". In: (2013). URL: http://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=1086.

[16] Brad Warner and Manavendra Misra. "Understanding Neural Networks as Statistical Tools". English. In: *The American Statistician* 50.4 (1996), pp. 284–293. ISSN: 00031305. URL: http://www.jstor.org/stable/2684922.