Machine Learning: An Introduction

Sayan Putatunda

Over the years, with the increase in storage capacity and the ease of vast amount of data collection, smart data analysis has become the order of the day. That is why "machine learning" has become one of the mainstays of the technology field over the past decade or so. This chapter aims to give an overview of the concepts of various supervised and unsupervised machine learning techniques such as support vector machines, k-nearest neighbor, artificial neural networks, random forests, cluster analysis, etc. Also, this chapter will give a brief introduction to deep learning, which is the latest fad in the analytics/data science industry.

1 Introduction

Machine learning (ML) can be described as the science and art by which computers learn things by themselves. We can always program computers to perform tasks and solve problems but for complex problems such as image recognition, speech recognition, etc., it is not possible to manually program all the aspects. So, we need the computer to learn by themselves and recognize different patterns (Domingos 2012). For example, let us consider the classic problem of spam detection. We can program to identify whether an incoming email is a spam or not just by writing conditions if the mail contains certain keywords. So, basically, this is similar to hardcoding or writing rules for all scenarios. But that is not scalable. So, a smarter way to operate would be by making the machines learn to distinguish between a spam and a non-spam using algorithms and feeding it historical email data. Here, "data" acts as a fuel. The machine can be used to then make predictions (i.e., spam/not spam) over unseen examples. This entire operation is achievable using machine learning. In

S. Putatunda (✉)
Indian Institute of Management Ahmedabad, Ahmedabad, India
e-mail: sayanp@iima.ac.in

A. K. Laha (ed.), *Advances in Analytics and Applications*, Springer Proceedings in Business and Economics, https://doi.org/10.1007/978-981-13-1208-3_1

short, machine learning is the phenomenon of enabling machines to learn and make predictions using past experiences (Başztanlar and Özuysal 2014).

The applications of machine learning are ubiquitous today. We can see it being used by companies such as Amazon, Apple, Google, Facebook, Microsoft, Uber, etc., in their products. Amazon Echo/Alexa is one such example. Another very recent example is the face detection feature in Apple iPhone X. Another interesting application of machine learning that companies like Google and Microsoft heavily uses is in Computational Advertising (Yang et al. 2017), which is basically targeted marketing where relevant advertisements are shown for different users when they input their queries in search engines such as Google or Bing. Some other examples of applications of machine learning include product recommendation, web search rankings, fraud detection, vehicle destination and travel time prediction, image processing, language translation, speech recognition and many more.

Machine learning is now being adapted across various industries, viz., banking, high tech, agriculture, manufacturing, Retail, etc. The different machine learning methods/algorithms can be broadly grouped under supervised and unsupervised learning methods. We will discuss these in the later sections. The "Data" can be either structured (i.e., tables, etc.) or unstructured (i.e., text, images, etc.). As far as tools for data science are concerned, SAS was the tool of choice for a longtime but ever since the open source revolution, R and Python are the tools of choice in most of the organizations and in academia.

2 Supervised Learning

In supervised learning, we have a target or response variable (i.e., y) for each independent or predictor variables (i.e., x_1, x_2,…, x_n). The goal is to build a model that can relate the y to x_i (where $i = 1, 2, \ldots, n$) with the objective of understanding the relationship between the target and the independent variables and also to make predictions of y for future observations of x_i (Gareth et al. 2013).

The various types of supervised learning problems can be divided into regression and classification problems. In case of regression, the target or response variable is continuous whereas it is categorical in case of classification. For example, prediction of house prices in a city given the features, viz., house area and household income is a regression problem whereas to predict whether an email is a spam or not given some predictor variables, is a classification problem. A common approach generally used for solving these kinds of problems is the linear models such as linear/logistic regression. However, in this section, we will focus on some of the more advanced methods such as k-nearest neighbor, artificial neural networks, support vector machines, and random forests. See Bishop (2006) and Hastie et al. (2009) for more details on linear models.

2.1 K-Nearest Neighbor (KNN)

The k-nearest neighbor algorithm can work in both regression and classification setting. It is basically an extension of the nearest neighbor rule (Cover and Hart 1967) that can be defined as a simple nonparametric classification technique. Here for a new observation say "M", its distance from other observations is calculated using a distance metric T, and then, M is classified based on the class/category of its nearest neighbor. Let us define this more formally. Suppose there are q training pairs $(x_1, c_1), \ldots, (x_q, c_q)$ where x_i is of dimension p and c_j indicates the class category of x_i. Let us consider M* be a test pattern with unknown class/category. If

$$T(M^*, x_u) = \min\{(T(M^*, x_i))\}$$

where i= 1,…, q, then M^* is assigned class c_u (Cover and Hart 1967; Murty and Devi 2011).

The above nearest neighbor rule is extended to KNN when the value of K is greater than 1. In such case, the modus operandi remains the same but instead of one nearest neighbor for a test observation M, we determine its k-nearest neighbors and the M is assigned the final class by taking a majority voting of the classes of all its k-nearest neighbors (Putatunda 2017). The same thing works in regression setup as well where instead of majority voting, we take an average of the target variable value of all the k-nearest neighbors. This method is known as KNN Regression (Navot et al. 2005).

The choice of the distance metric is a very important parameter since it can greatly impact the performance of the KNN model. Some well-known distance metrics used in the literature are Euclidean distance, Chebyshev distance, Manhattan distance, Mahalanobis distance, etc., (Weinberger and Saul 2009). Also, the KNN method is computationally expensive for large datasets. The time complexity for KNN algorithm on a dataset with n observations and p variables is O(np) (Kusner et al. 2014).

2.2 Artificial Neural Networks (ANN)

If we go through the history of evolution of neural networks, we can find that the origin of neural networks can be traced back to the 60s and the 80s when there were a lot of studies to understand and represent mathematically the various information processing in biological systems (Rosenblatt 1962; Rumelhart et al. 1986; Widrow and Hoff 1988). ANN is used for both classification and regression problems. In this chapter, we will focus on a type of ANN known as the "multilayer perceptrons"/"feedforward neural networks" (Bishop 2006).

A Feedforward neural network basically consists of a single or multiple hidden layers between an input layer and an output layer. And each of these layers comprises one or more neurons or nodes. Let us try to understand the structure of a basic neural network with two layers, i.e., a hidden layer and one output layer.

Suppose the input layer comprises T nodes corresponding to the input variables $x_1, \ldots, x_T$. Then

$$q_i^{(1)} = \sum_{j=1}^{T} w_{ij}^{(1)} x_i + w_{i0}^{(1)}$$

where i = 1,…, P and superscript (1) indicates that the mentioned parameters are in the first layer (i.e., the hidden layer). The above equation represents the P linear combinations of the input variables $x_1,\ldots,x_T$, where P is the number of hidden nodes that the hidden layer possesses. $w_{ij}^{(1)}$ is the weight, $w_{i0}^{(1)}$ is the bias, and finally, $q_i^{(1)}$ is known as the activations. Then, a nonlinear differentiable function f is used to transform these activations as shown below:

$$c_i = f\left(q_i^{(1)}\right)$$

where c_i is the output of the hidden nodes. The c_i's are further combined to obtain the output unit activations, i.e., $q_k^{(2)}$ as given in the following equation:

$$q_k^{(2)} = \sum_{s=1}^{Y} w_{ks}^{(2)} c_i + w_{k0}^{(2)}$$

where the superscript (2) denotes that the parameters belong to the second layer of the network (i.e., the output layer) and k = 1; N denotes the total number of outputs. Also, another activation function f^2 is used to transform $q_k^{(2)}$ to generate the network output, i.e., $y_k = f^2\left(q_k^{(2)}\right)$ (Bishop 2006; Laha and Putatunda 2017). Some choices for activation functions as given in the literature are Relu, tanh, etc., and the most common choice for the output activation function is the sigmoidal function.

In the literature, one can find that the approximation properties of the feedforward neural networks are well established. According to Hornik et al. (1989), multilayer feedforward neural networks can be described as "universal approximators".

2.3 *Support Vector Machines (SVM)*

The support vector machine was first proposed by Boser et al. (1992) as an optimal classification method. Vapnik (1995) later extended this for solving regression problems and thus proposed the method support vector regression (SVR). The basic idea of SVM is that often input data is not linearly separable in the normal feature space and SVM maps the input vectors x into a higher dimensional feature space H using the "kernel trick" (Cortes and Vapnik 1995). In the higher dimensional space H, the data becomes linearly separable (Hastie et al. 2009).

The support vector classifier can be represented as

$$g(\chi) = \alpha_0 + \sum_{i=1}^{n} \beta_i \langle \chi, \chi^* \rangle$$

where β_i and α_0 are the parameters. To estimate the parameters β_i and α_0, we need the $\langle \chi, \chi^* \rangle$, which represents the inner products among all the pairs of training observations as shown in the above equation (Gareth et al. 2013; Hastie et al. 2009). We then replace the inner product part of the above equation of a support vector classifier using a generalization, i.e., a function of the form $K(\chi_i, \chi_i^*)$, which we will refer to as "Kernel".

A Kernel is basically a function that is used to quantify the similarity between a pair of observations (Gareth et al. 2013). For example, a linear kernel as defined below computes the similarity between a pair of observations using Pearson correlation (Gareth et al. 2013).

$$K(\chi_i, \chi_i^*) = \sum_{j=1}^{N} \chi_{ij} \chi_{ij}^*$$

Another example of a very widely used kernel is the radial basis function (RBF) kernel (Gareth et al. 2013). The RBF kernel is defined as

$$K = (\chi_i, \chi_i^*) = \exp\left(-\zeta \sum_{j=1}^{N} (\chi_{ij} - \chi_{ij}^*)^2 \right)$$

where $\zeta > 0$ is a parameter.

As mentioned above, the modus operandi for SVM is basically combining kernels with support vector classifier. The space and time complexity of SVM are $O(n^2)$ and $O(n^3)$, respectively (Tsang et al. 2005). Thus, for large datasets, the computation time of SVM is quite high.

2.4 *Random Forest*

Before getting into the nitty-gritties of random forest, let us first try to understand decision trees and bagging. Decision trees are applicable to both regression and classification problems, i.e., when the target variables are continuous and categorical, respectively. The trees are generally upside down where the predictor space is split into different segments known as the *internal nodes*. The different nodes are connected through *branches*. Moreover, this is a greedy algorithm (Gareth et al. 2013). The major advantage of decision trees is that they are easy to explain but they are quite unstable in terms of predictive accuracy compared to that of other methods such as ANN, SVM, etc. Bagging is an ensemble learning method that was proposed by Breiman (1996). This is used to reduce the variance of a machine learning model.

Here, we take X bootstrapped samples from the training dataset and build our model M on each of the X samples. For final prediction, we either take an average of all the prediction (in case of regression) or take a majority voting of all the predictions (in case of classification).

As mentioned earlier, even though decision trees are easy to interpret, their performance in terms of predictive accuracy is not quite stable compared to that of other standard classification/regression methods. However, the predictive accuracy of trees can be increased by aggregating them, and random forest is one such approach. The random forest was first proposed by Breiman (2001). It is similar to bagged trees where decision trees are built on a number of bootstrapped samples of training dataset and the predictions are averaged/majority voting is taken to arrive at the final predictions. But the only difference here is that not all the predictors in the dataset, i.e., N are used for building trees on each bootstrapped samples. In fact, a random sample of P predictors is taken, and generally, the norm is $P = \sqrt{N}$ (Gareth et al. 2013).

3 Unsupervised Learning

In contrast to supervised learning, here we do not have a target or response variable for each independent variables (i.e., x_1, x_2,…, x_n). For example, a customer segmentation problem (i.e., suppose a marketing manager wants to understand who are his customers who are likely to purchase his new product based on features such as demographic characteristics, purchase behavior, etc.) would be in this category. A prominent method for unsupervised learning is the cluster analysis, which we will discuss in Sect. 3.1. A few other techniques that fall under the unsupervised learning category are the principal component analysis and the market basket analysis. One can refer Hastie et al. (2009) for more details.

3.1 Clustering

The basic idea of clustering is to partition the raw data or observations into different groups such that the observations in a group are similar to each other, i.e., homogeneous but each group has distinct characteristics (Gareth et al. 2013). So, for a customer segmentation problem as discussed earlier, a marketing manager can understand a customer better for a new product by grouping the customers into different clusters and profile them based on various characteristics. This will help the marketing manager to target the relevant customers through campaigns or advertisements.

The different clustering methods can be divided into (a) distance-based (e.g., hierarchical, K-means, etc.) and (b) density-based (e.g., DBSCAN, OPTICS, etc.). In this chapter, we will focus on two of the most widely used clustering methods, i.e., hierarchical and K-means clustering.

Hierarchical Clustering: In this approach, a hierarchy of clusters is created. The two approaches for building a hierarchical clustering model are (a) agglomerative—where observations merge to form clusters on their way up the hierarchy ("bottom-up approach"), and (b) divisive—where it begins with one cluster at the top and then it is broken down further as one goes down the hierarchy ("top-down approach") (Kaufman and Rousseeuw 1990). The final results are shown using a tree-like structure called "dendrogram". Moreover, a measure of dissimilarity is used between sets of observations to determine in which cluster they should belong (Kaufman and Rousseeuw 1990). A commonly used metric for dissimilarity is the "Euclidean distance".

K-Means Clustering: Here, we focus on partitioning the data into K nonoverlapping and distinct clusters. But the prerequisite here is that we need to specify the value of K. The goal of K-means clustering is to have a within-cluster variation as low as possible (Kaufman and Rousseeuw 1990).

The modus operandi is basically that we start by randomly assigning a number to observations from 1 to K and thus create initial clusters. We then iterate by computing cluster centroid for each K cluster and assign each observation to the nearest cluster. The nearness of an observation from the centroid of a cluster is determined using a distance metric such as the Euclidean distance (Kaufman and Rousseeuw 1990). The iteration stops when the cluster assignments do not change further.

4 Deep Learning

Deep learning is basically a form of "representation learning" where it is able to extract high-level features from raw data. In other words, the machine is able to develop complex concepts out of simpler ones (Goodfellow et al. 2016). We have explained the concept of feedforward neural network earlier in Sect. 2.2. A prominent example of deep learning would be a feedforward deep neural network (DNN) where we have many hidden layers comprising quite a few hidden nodes each. Each hidden layer is performing a mathematical transformation of the input and this helps uncover a new representation of the input data (Goodfellow et al. 2016).

Deep learning as an idea is not very new (since most of the algorithms have been there in existence since the 80s) but it is only after 2006 that it has shown great progress mainly due to the rise in computing power. However, the term "deep learning" is quite new. It is well established now that deep learning performs better with more data, and for some problems, it performs even better than some other sophisticated methods such as KNN, SVM, etc. It is especially in fields such as image processing and speech recognition where deep learning has got tremendous success.

Deep neural networks (DNNs) with greater depth can execute more instructions in sequence, and each layer encodes the necessary variation that explains the input data (Goodfellow et al. 2016). Apart from DNNs, there are other powerful deep learning algorithms such as autoencoders, recurrent neural networks (RNN), convolutional

neural networks (CNN), etc., that we have not discussed in detail here. An autoencoder is generally used to understand the representation of the data, and thus, it is a form of unsupervised learning. An interesting application of autoencoders is in dimensionality reduction. RNNs and CNNs have been very successful in sequence modeling and image processing, respectively. See Goodfellow et al. (2016) to explore the theoretical concepts of various deep learning algorithms in depth.

5 Conclusion

In this chapter, we have discussed the basic idea of machine learning and supervised/unsupervised learning. We give an overview of some of the well-known machine learning algorithms that fall into the category of supervised and unsupervised learning. We also give a brief introduction to deep learning. Moreover, most of the ML methods come under the category of supervised and unsupervised learning. However, there is one more category, i.e., reinforcement learning which we have not discussed in this chapter. Please refer to Sutton and Barto (1998) for an in-depth understanding of reinforcement learning.

We are now living in the age of Big Data (Manyika et al. 2011). There are some who believe that more data will lead to better model performance but it is always the choice of algorithms and its optimization that leads to models with better prediction accuracy and less execution time. However, while building models we need to ensure that the model should neither *overfit* (i.e., performs well in the training dataset but performs poorly on the unseen data that has not been used for model building) nor *underfit* (i.e., when the model performs poorly or gives bad predictions in the training dataset). This is also known as the "Bias–Variance Trade-off" (Hastie et al. 2009).

Machine learning is a vast and a very dynamic field. Almost every other day, we hear of innovations in this field either in terms of new methodological contributions or in terms of new applications of machine learning in different fields. In sum, these are very exciting times to be a data scientist or a machine learning researcher.

References

Başţanlar, Y., & Özuysal, M. (2014). Introduction to machine learning. *Methods in Molecular Biology, 1107,* 105–128.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. New York Inc: Springer.

Boser, B. E., Guyon, I. M., Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory, (COLT'92)* (pp. 144–152).

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*(421), 123–140.

Breiman, L. (2001). Random Forests. *Machine Learning, 45*(1), 5–32.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273–297.

Cover, P., & Hart, T. (1967). Nearest neighbour pattern classification. *IEEE Transactions on Information Theory*, *13*, 21–27.

Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM, 55*(10), 78.

Gareth, J., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning with applications in R*. Springer Texts in Statistics.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Hastie, T., Tibsharani, R., & Friedman, J. (2009). *The elements of statistical learning* (2nd ed.). Springer.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks, 2*(5), 359–366.

Kaufman, L., & Rousseeuw, P. J. (1990). *Finding groups in data: An introduction to cluster analysis* (1st ed.). New York: Wiley.

Kusner, M., Tyree, S., Weinberger, K. Q., & Agrawal, K. (2014). Stochastic neighbor compression. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)* (pp. 622–630).

Laha, A. K., Putatunda, S. (2017). Travel Time Prediction for GPS Taxi Data Streams. W.P.No. 2017-03-03, March 2017, Indian Institute of Management, Ahmedabad.

Manyika, J., Chui, M., & Brown, B. (2011). *Big data: The next frontier for innovation, competition, and productivity*. McKinsey Global Institute, June, p. 156.

Murty, M. N., & Devi, V. S. (2011). *Pattern recognition: an algorithmic approach*. Springer.

Navot, A., Shpigelman, L., Tishby, N., & Vaadia, E. (2006). Nearest neighbor based feature selection for regression and its application to neural activity. In *Advances in Neural Information Processing Systems 18 (Neural Information Processing Systems, NIPS 2005)* (pp. 996–1002).

Putatunda, S. (2017). *Streaming data: New models and methods with applications in the transportation industry*. Ph.D thesis, Indian Institute of Management Ahmedabad.

Rosenblatt, F. (1962). *Principles of neurodynamics: Perceptrons and the theory of brain mechanisms*. Washington: Spartan Books.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland, & P. Research-Group (Eds.), *Parallel distributed processing: explorations in the microstructure of cognition* (Vol. 1, pp. 318–362). Cambridge, MA, USA: MIT Press.

Sutton, R.,& Barto, A. (1998). *Reinforcement learning: An introduction*. MIT Press.

Tsang, I. W., Kwok, J. T., & Cheung, P.-M. (2005). Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research, 6,* 363–392.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer, New York Inc.

Weinberger, K. Q., & Saul, L. K. (2009). Distance metric learning for large margin nearest neighbor classication. *Journal of Machine Learning Research, 10,* 207–244.

Widrow, B., & Hoff, M. E. (1988). Adaptive switching circuits. In J. A. Anderson & E. Rosenfeld (Eds.), *Neurocomputing: Foundations of Research* (pp. 123–134). Cambridge, MA, USA: MIT Press.

Yang, Y., Yang, Y., Jansen, B. J., & Lalmas, M. (2017). Computational advertising: A paradigm shift for advertising and marketing? *IEEE Intelligent Systems, 32*(3), 3–6.