International Transaction of Electrical and Computer Engineers System, 2017, Vol. 4, No. 2, 55-61
Available online at http://pubs.sciepub.com/iteces/4/2/2
©Science and Education Publishing
DOI:10.12691/iteces-4-2-2

Data Mining, Machine Learning and Big Data Analytics

Lidong Wang*

Department of Engineering Technology, Mississippi Valley State University, Itta Bena, MS, USA
*Corresponding author: lwang22@students.tntech.edu

Abstract This paper analyses deep learning and traditional data mining and machine learning methods; compares the advantages and disadvantage of the traditional methods; introduces enterprise needs, systems and data, IT challenges, and Big Data in an extended service infrastructure. The feasibility and challenges of the applications of deep learning and traditional data mining and machine learning methods in Big Data analytics are also analyzed and presented.

Keywords: *big data, Big Data analytics, data mining, machine learning, deep learning, information technology, data engineering*

Cite This Article: Lidong Wang, “Data Mining, Machine Learning and Big Data Analytics.” *International Transaction of Electrical and Computer Engineers System*, vol. 4, no. 2 (2017): 55-61. doi: 10.12691/iteces-4-2-2.

1. Introduction

Data mining focuses on the knowledge discovery of data. Machine learning concentrates on prediction based on training and learning. Data mining uses many machine learning methods; machine learning also uses data mining methods as pre-processing for better learning and accuracy. Machine learning includes both supervised and unsupervised learning methods. Data mining has six main tasks: clustering, classification, regression, anomaly or outlier detection, association rule learning, and summarization. The feasibility and challenges of the applications of data mining and machine learning in big data has been a research topic although there are many challenges. Data dimension reduction is one of the issues in processing big data.

High-dimensional data can cause problems for data mining and machine learning although high-dimensionality can help in certain situations, for example, nonlinear classification. Nevertheless, it is important to check whether the dimensionality can be reduced while preserving the essential properties of the full data matrix. [1]. Dimensionality reduction facilitates the classification, communication, visualization, and storage of high-dimensional data. The most widely used method in dimensionality reduction is principal component analysis (PCA). PCA is a simple method that finds the directions of greatest variance in the dataset and represents each data point by its coordinates along each of these directions [2]. The direction with the largest projected variance is called the first principal component. The orthogonal direction that captures the second largest projected variance is called the second principal component, and so on [1]. PCA is useful when there are a large number of variable within the data, and there is some redundancy in those variables. In this situation, redundancy means that some of the variables are correlated with one another. Because of this redundancy, PCA can be used to reduce the observed variables into a smaller number of principal components [3].

Factor analysis is another method for dimensionality reduction. It is useful for understanding the underlying reasons for the correlations among a group of variables. The main applications of factor analysis are reducing the number of variables and detecting structure in the relationships among variables. Therefore, factor analysis is often used as a structure detection or data reduction method. Specifically, it is used to find the hidden factors behind observed variables and reduce the number of intercorrelated variables. In factor analysis, it is assumed that some unobservable latent variables generate the observed data. The data is assumed to be a linear combination of the latent variables and some noise. The number of latent variables is possibly less than the number of variables in the observed data, which fulfils the dimensionality reduction [4,5].

In practical applications, the proportions of 75% and 25% are often used for the training and validation datasets, respectively. However, the most frequently used method, especially in the field of neural networks, is dividing the data set into three blocks: training, validation, and testing. The testing data will not be used in the modelling phase [6]. The *k*-fold cross-validation technique is a common technique that is used to estimate the performance of a classifier because it overcomes the problem of over-fitting [7]. In *k*-fold cross-validation, the initial data is randomly partitioned into *k* mutually exclusive subsets or “folds”. Training and testing are performed *k* times. Each sample is used the same number of times for training and once for testing [8]. Normalization is particularly useful for classification algorithms involving neural networks, or distance measurements such as nearest-neighbor classification and clustering. For distance-based methods, normalization helps prevent attributes with initially large ranges (e.g., income) from outweighing the attributes with initially smaller ranges (e.g., binary attributes). There are many methods for data normalization such as min-max normalization, z-score normalization, and normalization by decimal scaling.

The purposes of this paper are to 1) analyze deep learning and traditional data mining and machine learning methods (including *k*-means, *k*-nearest neighbor, support vector machines, decision trees, logistic regression, Naive Bayes, neural networks, bagging, boosting, and random forests); 2) compares the advantages and disadvantage of the traditional methods; 3) introduces enterprise needs, systems and data, IT challenges, and Big Data in an extended service infrastructure; and 4) discuss the feasibility and challenges of the applications of deep learning and traditional data mining and machine learning methods in Big Data analytics.

2. Some Methods in Data Mining and Machine Learning

2.1. *k*-means, *k*-modes, *k*-prototypes and Clustering Analysis

Clustering methods can be classified into the following categories: partitioning method, hierarchical method, model-based method, grid-based method, density-based method, and the constraint-based method. The main advantage of clustering over classification is its adaptability to changes and helping single out useful features that distinguish different groups [9]. A good clustering method produces high quality clusters with high intra-class similarity and low inter-class similarity. The quality of clustering depends upon the appropriateness of the method for the dataset, the (dis)similarity measure used, and its implementation. The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns. Types of data in clustering analysis include nominal (categorical), interval-scaled variables, binary variables, ordinal variables, and mixed types [10].

k-means uses a greedy iterative approach to find clustering that minimizes the sum of squared errors (SSE). It possibly converges to a local optimum instead of a globally optimum [1]. Important properties of the *k*-means algorithm include [11]: 1) efficient in processing large data sets; 2) works only on numerical values; 3) clusters have convex shapes. Users need to specify *k* (the number of clusters) in advance. The method possibly terminates at a local optimum. The global optimum may be found using techniques such as deterministic annealing and genetic algorithms. The *k*-means method is not applicable for categorical data while *k*-modes is a method for categorical data that uses modes. *k*-modes use new dissimilarity measures to deal with categorical objects and use a frequency-based method to update the modes of clusters. The *k*-prototypes method can deal with a mixture of categorical and numerical data [10].

2.2. *k*-Nearest Neighbors

k-nearest neighbor (*k*-NN) classification finds a group of *k* objects in the training set that are closest to the test object and bases the assignment of a label on the predominance of a particular class in this neighborhood. *k*-NN involves assigning an object a class of its nearest neighbor or of the majority of its nearest neighbors. Specifically speaking, the *k*-NN classification finds the *k* training instances that are closest to the unseen instance and takes the most commonly occurring classification for these *k* instances. There are several key issues that affect the performance of *k*-NN. One is the choice of *k*. If *k* is too small, the result can be sensitive to noise points. On the other hand, if *k* is too large, the neighborhood may include too many points from other classes. An estimate of the best value for *k* can be obtained by cross-validation. Given enough samples, larger values of *k* are more resistant to noise [12,13]. The *k*-NN algorithm for classification is a very simple ‘instance-based’ learning algorithm. Despite its simplicity, it can offer very good performance on some problems [3]. Important properties of *k*-NN algorithm are [11]: 1) it is simple to implement and use; 2) it needs a lot of space to store all objects.

2.3. Support Vector Machine

Support vector machines (SVM) is a supervised learning method used for classification and regression tasks [3]. SVM has been found to work well on problems that are sparse, nonlinear, and high-dimensional. An advantage of the method is that building the model only uses support vectors rather than the whole training dataset. Hence, the size of the training set is usually not a problem. Also, the model is less affected by outliers due to only using the support vectors to build it. A disadvantage is that the algorithm is sensitive to the choice of tuning option (e.g., the type of transformations to perform). This makes it time-consuming and harder to use for the best model. Another disadvantage is that the transformations are performed during both building the model and scoring new data. This makes it computationally expensive. SVM works with numeric and nominal values; the SVM classification supports both binary and multiclass targets [14].

2.4. Trees and Logistic Regression

Decision trees used in data mining include two main types: 1) classification trees for predicting the class which the data belongs to; and 2) regression trees for predicting the outcome that is a real number. Classification trees and regression trees provide different approaches to prediction [15]. When constructing a tree, measures such as statistical significance, information gain, Gini index, and so on can be used to assess the performance of a split. When a decision tree is built, many of the branches will reflect anomalies in the training data due to noise or outliers. Tree pruning methods address this problem of overfitting the data. Pruned trees tend to be smaller and less complex, thus easier to comprehend. They are usually faster and better at correctly classifying independent test data [8]. There are two approaches to prune a tree: 1) pre-pruning — the tree is pruned by halting its construction early; 2) post-pruning — this approach removes a sub-tree from a fully grown tree [9]. A strategy of post-pruning (sometimes called backward pruning) rather than pre-pruning (or forward pruning) is often adopted after building a complete tree [16]. Both recursive partitioning trees and conditional inference trees are nonparametric, work on both classification and regression problems, and are very flexible and easy to interpret while they are prone to over-fitting. Conditional inference trees are less prone to bias than a recursive partitioning tree [7]. Logistic regression is a regression model where the dependent variable is categorical. It is computationally inexpensive, easy to implement, good in knowledge representation, and easy to interpret. However, it is prone to underfitting and may have low accuracy [5].

2.5. Naïve Bayes

The Naïve Bayes classifier is a method of classification that does not use rules, a decision tree or any other explicit representation of the classifier. Rather, it uses the probability theory to find the most possible classifications [13]. Naïve Bayes works with a small amount of data and nominal values [5]. Important properties of the Naive Bayes algorithm are [11]: 1) it is very easy to construct and training is also easy and fast; and 2) it is highly scalable.

The Naive Bayes classifier's beauty is in its simplicity, computational efficiency, good classification performance. In fact, it often outperforms more sophisticated classifiers even when the underlying assumption of independent predictors is far from true. This advantage is especially for the situation when the number of predictors is very large. There are more features about Naive Bayes. First, the Naive Bayes classifier requires a very large number of records to obtain good results. Second, where a predictor category is not present in the training data, Naive Bayes assumes that a new record with that category of the predictor has zero probability. This can be a problem if this rare predictor value is important. Finally, good performance is obtained when the goal is classification or ranking of records according to their probability of belonging to a certain class. However, when the goal is to actually estimate the probability of class membership, this method provides very biased results. For this reason, the Naive Bayes method is rarely used in credit scoring [17].

2.6. Neural Networks

Neural networks, also called artificial neural networks, are models for classification and prediction [17]. Neural network algorithms are inherently parallel. Parallelization methods can be used to speed up the computation process. In addition, several techniques have recently been developed for the extraction of rules from trained neural networks. This contributes to the application of neural networks for classification and prediction in data mining [6]. Important properties of neural networks are as follows [17]:

- First, although neural networks are capable of generalizing from a set of examples, extrapolation is still a serious danger. If the network sees only cases in a certain range, then its predictions outside this range can be completely invalid.
- Second, neural networks do not have a built-in variable selection mechanism. This means that there is need for careful consideration of predictors. Combination with classification and regression trees and other dimension reduction techniques (e.g., principal components analysis) is often used to identify key predictors.
- Third, the extreme flexibility of the neural network relies heavily on having sufficient data for training purposes. A neural network performs poorly when the training set size is insufficient, even if the relationship between the response and predictors is very simple.
- Fourth, a technical problem is the risk of obtaining weights that lead to a local optimum rather than the global optimum.
- Finally, neural networks are involved in much computation and require longer runtime than other classifiers. The run time increases greatly when the number of predictors grows.

The most popular neural network algorithm is backpropagation. Backpropagation uses a method of gradient descent. The target value may be the known class label of the training tuple (for classification problems) or a continuous value (for prediction) [6]. The tradeoff should be between under- and over-fitting to decide the size of hidden layer. Using too few nodes might not be sufficient to capture complex relationships. On the other hand, too many nodes may result in overfitting. A rule of thumb is to start with *p* (number of predictors) nodes and gradually decrease/increase a bit while checking for overfitting [17].

Advantages of neural networks include their good predictive performance, tolerance of noisy data as well as their ability to classify patterns on which they have not been trained. They can be used when you may have little knowledge of the relationships between attributes and classes. They are well-suited for continuous-valued inputs and outputs, unlike most decision tree algorithms [6,17]. Neural networks are very general and can approximate complicated relationships. Their weakest point is in providing insight into the structure of the relationship, and hence their “black-box" reputation. The user of neural networks must make many modelling assumptions, such as the number of hidden layers and the number of units in each hidden layer, and usually there is little guidance on how to do this. Furthermore, back-propagation can be quite slow if the learning constant is not chosen correctly [17,18].

Reducing the data dimensionality can be performed with neural networks. High-dimensional data can be converted to low-dimensional codes by training a multilayer neural network with a small central layer to reconstruct high-dimensional input vectors. Gradient descent can be used for fine-tuning the weights in such ‘‘autoencoder’’ networks, but this works well only if the initial weights are close to a good solution. An effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes was proposed. It works better than principal components analysis as a tool to reduce the dimensionality of data [2].

2.7. Deep Learning

Deep Learning is a new area in machine learning research, which has been introduced with the objective of moving machine learning closer to one of its original goals — artificial intelligence. Deep Learning is about learning multiple levels of representation and abstraction that help to make sense of data [19]. Deep machines are more efficient for representing certain classes of functions; particularly for those involved in visual recognition, they can represent more complex functions with less “hardware”. SVMs and Kernel methods are not deep. Classification trees are not deep either because there are no hierarchy of features. Deep learning involves non-convex loss functions and deep supervised learning is non-convex [20]. Deep learning has the potential in dealing with big data although there are challenges.

Some methods have been proposed for using unlabeled data in deep neural network-based architectures. These methods either perform a greedy layer-wise pre-training of weights using unlabeled data alone followed by supervised fine-turning, or learn unsupervised encodings at multiple levels of architecture jointly with a supervised signal. For the latter, the basic setup is as follows: 1) choose an unsupervised learning algorithm; 2) choose a model with a deep architecture; 3) the unsupervised learning is plugged into any (or all) layers of the architecture as an auxiliary task; and 4) train supervised and unsupervised tasks using the same architecture simultaneously [21].

2.8. Comparison of Different Methods and Ensemble Methods

Table 1 compares the advantages and disadvantages of traditional data mining (DM) and machine learning (ML) methods.

Ensemble methods increase the accuracy of classification or prediction. Bagging, boosting, and random forest are the three most common methods in ensemble learning. The bootstrap (or bagged) classifier is often better than a single classifier that is derived from the original training set. The increased accuracy occurs because the composite model reduces the variance of the individual classifiers. For prediction, a bagged predictor improves the accuracy over a single predictor. It is robust to overfitting and noisy data. Bootstrap methods can be used not only to assess a model’s discrepancy, but also improve the accuracy. Bagging and boosting methods use a combination of models and combine the results of more than one method. Both bagging and boosting can be used for classification as well as prediction [6,7,8,18].

Table 1. Advantages and Disadvantages of Traditional DM/ML Methods

Algorithms	Advantages	Disadvantages
The *k-means* method [22,23]	• Relatively efficient • Can process large data sets.	• Often terminates at a local optimum. • Applicable only when mean is defined. • Not applicable for categorical data. • Unable to handle noisy data. • Not suitable to discover clusters with non-convex shapes.
k-nearest neighbor (*k*-NN) classifier [7,15]	• Nonparametric • Zero cost in the learning process • Classifying any data whenever finding similarity measures of any given instances • Intuitive approach • Robust to outliers on the predictors	• Expensive computation for a large dataset • Hard to interpret the result • The performance relies on the number of dimensions • Lack of explicit model training • Susceptible to correlated inputs and irrelevant features • Very difficult in handling data of mixed types.
Support vector machine (SVM) [5,15,22]	• Can utilize predictive power of linear combinations of inputs • Good prediction in a variety of situations • Low generalization error • Easy to interpret results	• Weak in natural handling of mixed data types and computational scalability • Very black box • Sensitive to tuning parameters and kernel choice • Training an SVM on a large data set can be slow • Testing data should be near the training data
Decision Trees [7,15]	• Some tolerance to correlated inputs. • A single tree is highly interpretable, • Can handle missing values. • Able to handle both numerical and categorical data. • Performs well with large datasets.	• Cannot work on (linear) combinations of features. • Relatively less predictive in many situations. • Practical decision-tree learning algorithms cannot guarantee to return the globally-optimal decision tree. • Decision-tree can lead to overfitting.
Logistic regression [7]	• Provides model logistic probability • Easy to interpret • Provides confidence interval • Quickly update the classification model to incorporate new data	• Does not handle the missing value of continuous variables • Suffers multicollinearity • Sensitive to extreme values of continuous variables
Naïve Bayes [5,7]	• Suitable for relative small training set • Can easily obtain the probability for a prediction • Relatively simple and straightforward to use • Can deal with some noisy and missing data • Can handles multiple classes	• Prone to bias when increasing the number of training sets • Assumes all features are independent and equally important, which is unlikely in real-world cases. • Sensitive to how the input data is prepared.
Neural networks [15]	• Good prediction generally • Some tolerance to correlated inputs • Incorporating the predictive power of different combinations of inputs	• Not robust to outliers • Susceptible to irrelevant features • Difficult in dealing with big data with complex model

Bagging, which stands for bootstrap aggregation, is an ensemble classification method that uses multiple bootstrap samples (with replacement) from the input training data to create slightly different training sets [1]. Bagging is the idea of collecting a random sample of observations into a bag. Multiple bags are made up of randomly selected observations obtained from the original observations from the training dataset [14]. Bagging is a voting method of using bootstrap for different training sets and using the training sets to make different base learners. The bagging method employs a combination of base learners to make a better prediction [7].

Boosting is also an ensemble method which attempts to build better learning algorithms by combining multiple more simple algorithms [24]. Boosting is similar to the bagging method. It first constructs the base learning in sequence, where each successive learner is built for the prediction residuals of the preceding learner. With the means to create a complementary learner, it uses the mistakes made by previous learners to train the next base learner. Boosting trains the base classifiers on different samples [1,7]. Boosting can fail to perform if there is insufficient data or if the weak models are overly complex. Boosting is also susceptible to noise [14]. The most popular boosting algorithm is AdaBoost that is “adaptive.” AdaBoost is extremely simple to use and implement (far simpler than SVMs), and often gives very effective results [24]. AdaBoost works with numeric values and nominal values. It has low generalization error, is easy to code, works with most classifiers, and has no parameters to adjust. However, it is sensitive to outliers [5].

Although bagging and randomization yield similar results, it sometimes pays to combine them because they introduce randomness in different and perhaps complementary ways. A popular algorithm for learning random forests builds a randomized decision tree in each iteration of the bagging algorithm and often produces excellent predictors [16]. The random forests method is a tree-based ensemble approach that is actually a combination of many models [1,15]. It is an ensemble classifier that consist of many decision trees [25]. A random forest grows many classification trees, obtaining multiple results from a single input. It uses the majority of votes from all the decision trees to classify data or use an average output for regression [7].

Random forest models are generally very competitive with nonlinear classifiers such as artificial neural nets and support vector machines. A random forest model is a good choice for model building because of very little pre-processing of the data, no requirement for data normalization, and being resilient to outliers. The need for variable selection is avoided because the algorithm effectively does its own. Because many trees are built using two levels of randomness (observations and variables), each tree is effectively an independent model. The random forest algorithm builds multiple decision trees using a concept called bagging to introduce random sampling into the whole process. In building each decision tree, the random forest algorithm generally does not perform any pruning of the decision tree. Overfitted models tend not to perform well on new data. However, a random forest of overfitted trees can deliver a very good model that performs well on new data [14].

3. Big Data in Service Infrastructure and IT Challenges

As enterprise data challenges continue to grow (see Table 2 [26]), traditional technologies have challenges in handling unstructured, Cloud, and Big Data sources. Table 3 [27] shows Big Data as part of a virtualized service infrastructure. Hardware infrastructure is virtualized with cloud computing technologies; On top of this cloud-based infrastructure, Software as a Service (SaaS); and on top of SaaS, Business Processes as a Service (BPaaS) can be built. In parallel, Big Data will be offered as a service and embedded as the precondition for Knowledge services, e.g., the integration of Semantic Technologies for the analysis of unstructured and aggregated data. Big Data as a Service can be treated as an extended layer between PaaS and SaaS. Knowledge workers or data scientists are needed to run Big Data and Knowledge.

Table 2. Enterprise Needs, Systems and Data, and IT Challenges

Business Needs	Systems and Data	IT Challenges
• Access all information of value • Business capability and value driven • Virtualized & unified semantic business views of data • Fast, iterative, self-service, pervasive • Right information to right user at right time	• Inventory System (MS SQL Server) • Billing System (Web Service-Rest) • Customer Relationship Management (CRM) (MySQL) • Big Data, Cloud (Hadoop, Web) • Customer Voice (Internet, Unstructured) • Product Catalog (Web Service-SOAP) • Product Data (CSV) • Log Files (.txt/.log files)	• Data silos • Exponential data growth • Unstructured, Web & Big Data • IT complexity, rigidity • Inherent latency • Move to Cloud • High costs

Table 3. Big Data in an Extended Service Infrastructure

Layers	Services
Layer 1	Business Process as a Service (BPaaS), Knowledge as a Service (KaaS)
Layer 2	Software as a Service (SaaS), Big Data as a Service (BDaaS)
Layer 3 (Cloud Infrastructure)	Platform as a Service (PaaS)
Layer 4 (Cloud Infrastructure)	Infrastructure as a Service (IaaS)

4. Data Mining and Machine Learning in Big Data Analytics

Hadoop is a tool of Big Data analytics and the open-source implementation of MapReduce. The following brief list identifies the MapReduce implementations of three algorithms [5]:

- Naïve Bayes—This is one of a few algorithms that is naturally implementable in MapReduce. It's easy to calculate sums in MapReduce. Given a class, the probability of a feature can be calculated in Naïve Bayes method, the results from a given class can be given to an individual mapper, the Reducer can be used to sum up the results.
- Support vector machines (SVMs) —There's also an approximate version of SVM called proximal SVM which computes a solution much faster and is easily used in a MapReduce framework.
- Singular value decomposition—The Lanczos algorithm is an efficient method for approximating eigenvalues. This algorithm can be used in a series of MapReduce jobs to efficiently find the singular values in a large matrix.

However, the above three methods cannot be used in Big Data analytics. Traditional machine learning (ML) techniques are unsuitable for big data classification because: (1) An ML technique that is trained on a particular labelled datasets or data domain may not be suitable for another dataset or data domain; (2) an ML technique is in general trained using a certain number of class types and a large varieties of class types found in dynamically growing big data; and (3) an ML technique is developed based on a single learning task, and thus they are not suitable for multiple learning tasks and knowledge transfer requirements of Big data analytics [28]; and (4) memory constraint is a challenge. Although algorithms typically assume that training data samples exist in main memory, big data does not fit into it [29].

Big data mining is more challenging compared with traditional data mining algorithms. Taking clustering as an example, a natural way of clustering big data is to extend existing methods (such as k-means) so that they can cope with the huge workloads. Most extensions usually rely on analyzing a certain number of samples of big data, and vary in how the sample-based results are used to derive a partition for the overall data [30]. The k-NN classifiers do not construct any classifier model explicitly; instead they keep all training data in memory. Hence, they are not amenable to big data applications [31]. Splitting criteria of decision trees are chosen based on some quality measures such as information gain which requires handling the entire data set of each expanding nodes. This makes it difficult for decision trees to be applied to big data applications. Support vector machine (SVM) shows good performance for data sets in a moderate size. It has inherent limitations to big data applications [31].

Deep machine learning has the potential in dealing with big data. However, it has some challenges in big data applications because it requires significant amount of training time [31,32]. Deep learning challenges in Big Data analytics lie in: incremental learning for non-stationary data, high-dimensional data, and large-scale models [32]. The Variety characteristic of Big Data analytics, focuses on the variation of the input data types and domains in big data. Domain adaptation during learning is an important focus of study in deep learning, where the distribution of the training data is different from the distribution of the test data. In some big data domains, e.g., cyber security, the input corpus consists of a mix of both labelled and unlabeled data. In such cases, deep learning algorithms can incorporate semi-supervised training methods towards the goal of defining criteria for good data representation learning [33].

Representation-learning algorithms help supervised learning techniques to achieve high classification accuracy with computational efficiency. They transform the data, while preserving the original characteristics of the data, to another domain so that the classification algorithms can improve accuracy, reduce computational complexity, and increase processing speed. However, Big Data classification requires multi-domain, representation-learning (MDRL) technique because of its large and growing data domain. The MDRL technique includes feature variable learning, feature extraction learning, and distance-metric learning. Several representation-learning techniques have been proposed in the machine learning research. The recently proposed cross-domain, representation-learning (CDRL) technique maybe suitable for the big data classification along with the suggested network model; however, the implementation of the CDRL technique to big data classification will encounter several challenges, including the difficulty in selecting relevant features, constructing geometric representation, extracting suitable features, and separating various types of data. Also, the continuity parameter of big data introduces the problems that need to be addressed by lifelong learning techniques. The learning of big data characteristics in short term may not be suitable for long-term. Hence the machine lifelong learning (ML3) techniques should be used. The concept of ML3 provides a framework that can retain learned knowledge with training examples throughout the learning phases [31].

5. Conclusions

Dimensionality reduction can aid data visualization. PCA is the most commonly used technique for dimensional reduction. Factor analysis can be used as a data reduction or structure detection method. The k-means method is relatively efficient, but it possibly terminates at a local optimum.

k-NN is simple to implement and robust to outliers on the predictors; however, it is very difficult for it to handle data with mixed types. SVM works well on problems that are sparse, nonlinear, and high-dimensional; but it is weak in natural handling of mixed data types and computational scalability. Decision trees performs well with large datasets, but can lead to overfitting. Tree pruning is performed to remove anomalies in the training data due to noise or outliers. Logistic regression is computationally inexpensive, but it is prone to underfitting and may have low accuracy. The Naive Bayes algorithm is easy to construct and training is fast; it is suitable for relative small training set and prone to bias. Neural networks have good predictive performance and tolerance of noisy data;

however, it is very difficult for the method to deal with big data with complex models. Bagging, boosting, and random forests are the three most common ensemble methods that use a combination of models to increase accuracy.

Traditional technologies have challenges in handling unstructured and big data sources. Big Data as a Service (BDaaS) can be an extended layer in the service infrastructure. Traditional data mining and machine learning (ML) techniques such as *k*-means, *k*-NN, decision trees, and SVM are unsuitable for handling big data. Deep learning has the potential in dealing with big data although there are challenges.

References

[1] Zaki MJ, Meira Jr W, Meira W. Data mining and analysis: fundamental concepts and algorithms. Cambridge University Press; 2014 May 12.

[2] Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. science. 2006 Jul 28; 313(5786): 504-507.

[3] Wikibook, Data Mining Algorithms In R - Wikibooks, open books for an open world. PDF generated using the open source mwlib toolkit. See http://code.pediapress.com/, 2014 14 Jul.

[4] Jackson J. Data Mining; A Conceptual Overview. Communications of the Association for Information Systems. 2002 Mar 22; 8(1): 19.

[5] Harrington P. Machine learning in action. Greenwich, CT: Manning; 2012 Apr 16.

[6] Paolo G. Applied data mining: statistical methods for business and industry. John Wiley & Sons Ltd, 2003.

[7] Yu-Wei CD. Machine learning with R cookbook. Packt Publishing Ltd; 2015, Mar 26.

[8] Han J, Pei J, Kamber M. Data mining: concepts and techniques. Elsevier; 2011 Jun 9.

[9] Tutorialspoint. Data Mining: data pattern evaluation, Tutorials Point (I) Pvt. Ltd, 2014.

[10] Andreopoulos B. Literature Survey of Clustering Algorithms, Workshop, Department of Computer Science and Engineering, York University, Toronto, Canada, 2006 June 27.

[11] Sharma S and Gupta RK. Intrusion Detection System: A Review. International Journal of Security and Its Applications. 2015, 9(5): 69-76.

[12] Kumar V, Wu X, editors. The top ten algorithms in data mining. CRC Press; 2009.

[13] Bramer M. Principles of data mining. London: Springer; 2007 Mar 6.

[14] Williams G. Data mining with Rattle and R: The art of excavating data for knowledge discovery. Springer Science & Business Media; 2011 Aug 4.

[15] Clark M. An introduction to machine learning: with applications in *R*. University of Notre Dame, USA, 2013.

[16] Witten IH, Frank E, Hall MA, Pal CJ. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann; 2016 Oct 1.

[17] Galit S, Nitin P, Peter B. Data Mining In Excel: Lecture Notes and Cases. Resampling Stats, Inc., USA, 2005 December 30.

[18] Ledolter J. Data mining and business analytics with R. John Wiley & Sons; 2013 May 28.

[19] LISA Lab. Deep Learning Tutorial. University of Montreal, Canada, 2015 September.

[20] LeCun Y, Ranzato M. Deep learning tutorial. InTutorials in International Conference on Machine Learning (ICML'13) 2013 Jun.

[21] Weston J, Ratle F, Mobahi H, Collobert R. Deep learning via semi-supervised embedding. InNeural Networks: Tricks of the Trade. Springer Berlin Heidelberg 2012, 639-655.

[22] Andreopoulos B. Literature Survey of Clustering Algorithms, Workshop, Department of Computer Science and Engineering, York University, Toronto, Canada, 2006 June 27.

[23] Sharma S and Gupta RK. Intrusion Detection System: A Review. International Journal of Security and Its Applications. 2015, 9(5): 69-76.

[24] Hertzmann A, Fleet D. Machine Learning and Data Mining Lecture Notes. Computer Science Department, University of Toronto. 2010.

[25] Karatzoglou A. Machine Learning in R. Workshop, Telefonica Research, Barcelona, Spain. 2010 December 15.

[26] Viña A. Data Virtualization Goes Mainstream, White Paper, Denodo Technologies, 2015.

[27] Curry E, Kikiras P, Freitas A. et al. Big Data Technical Working Groups, White Paper, BIG Consortium, 2012.

[28] Suthaharan S., Big Data Classification: Problems and Challenges in Network Intrusion Prediction with Machine Learning, *Performance Evaluation Review*, 41 (4), March 2014, 70-73.

[29] S. Hido, S. Tokui, S. Oda, Jubatus: An Open Source Platform for Distributed Online Machine Learning, Technical Report of the Joint Jubatus project by Preferred Infrastructure Inc., and NTT Software Innovation Center, Tokyo, Japan, NIPS 2013 Workshop on Big Learning, Lake Tahoe. December 9, 2013. Pp. 1-6.

[30] C.L. P. Chen, C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data," *Information Sciences*, Vol. 275, No. 10, pp. 314-347, 2014.

[31] K. M. Lee, "Grid-based Single Pass Classification for Mixed Big Data," *International Journal of Applied Engineering Research*, Vol. 9, No. 21, pp. 8737-8746, 2014.

[32] M. M. Najafabadi, F. Villanustre, T. M Khoshgoftaar, N. Seliya, R. Wald and E. Muharemagic, Deep learning applications and challenges in big data analytics, *Journal of Big Data,* 2 (1), 2015.

[33] Najafabadi MM, Villanustre F, Khoshgoftaar TM, Seliya N, Wald R, Muharemagic E. Deep learning applications and challenges in big data analytics. Journal of Big Data. 2015 Feb 24; 2(1):1.