Journal of Theoretical and Applied Information Technology
29th February 2020. Vol.98. No 04

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# SUPERVISED MACHINE LEARNING FOR SMART DATA ANALYSIS IN INTERNET OF THINGS ENVIRONMENT: AN OVERVIEW

**MOHAMMED H. ALSHARIF [1,*], WILLIAM A. MOSIER[2], OSAMA AHMAD ALOMARI[3], KHALID YAHYA[4]**

[1]Department of Electrical Engineering, College of Electronics and Information Engineering, Sejong University, 209 Neungdong-ro, Gwangjin-gu, Seoul 05006, Korea

[2]Department of Biomedical Sciences, School of Health Sciences, Istanbul Gelisim University, Istanbul, Turkey

[3]Department of Computer Engineering, Faculty of Engineering and Architecture, Istanbul Gelisim University, Istanbul, Turkey

[4]Department of Mechatronics Engineering, Istanbul Gelisim University, Avcılar, 34310, İstanbul, Turkey

E-mail: [1]moh859@gmail.com, [2]drwilliammosier@gmail.com, [3]oalomari@gelisim.edu.tr, [4]koyahya@gelisim.edu.tr

**ABSTRACT**

Machine learning techniques will contribution to making Internet of Things (IoT) applications that are considered the most significant sources of new data in the coming future more intelligent, where the systems will be able to access raw data from different resources over the network and analyze this information in order to extract knowledge. This study focuses on supervised machine learning techniques that is considered the main pillar of the IoT smart data analysis. This study includes reviews and discussions of substantial issues related to supervised machine learning techniques, highlighting the advantages and limitations of each algorithm, and discusses the research trends and recommendations for further study.

**Keywords:** *Machine learning; Artificial intelligence; Supervised learning; Big data; Internet of Things.*

## 1. INTRODUCTION

Big data can be defined as high variety, high-velocity, and high-volume data. Big data requires cost-effective and innovative procedures of information processing to empower greater insight, problem-solving, and process automation [1]. Internet of Things (IoT) is considered the most significant source of new data, since IoT creates a platform in which physical objects can mimic the specific human sensory capabilities of perception, vision, hearing, and thinking. Buoyed with these human sensory capabilities and the emerging tactile Internet, machines can communicate with one another, share relevant information and make real-time decisions with minimal human input [2]. Systems need the ability to access raw data, coming from different resources within a network and analyze the data to extract useful information. Machine learning (ML) technology is a specific type of algorithm that can be applied to many different domains, data types, and data models [3]. Accordingly, ML is seen as providing a significant contribution to making IoT applications more intelligent [4].

ML is a type of Artificial Intelligence (AI) that provides machines with the ability to learn pattern recognition [5]. A learning algorithm depends on a set of samples as an input that name a training set; thus, learning algorithms can be classified into three main categories of learning (Figure 1) [6]:

1. Supervised learning. This learning algorithm uses samples of input vectors and their target vectors. The target vectors are typically referred to as labels. This type of learning attempts to predict the output vector for a specified input vector. Applications that have target labels

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

contain a finite quantity of distinct categories. This is, typically, referred to as *classification tasks*. When these target labels are composed of one or more constant variables they are called *regression tasks* [5].

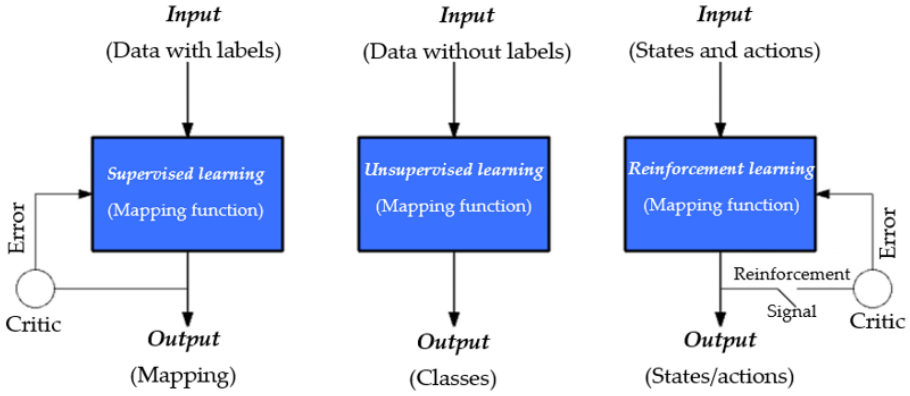


*Figure 1: Classification of learning models.*

2. Unsupervised learning. This learning algorithm does not require labeling of the training set. The objective of this type of learning is to identify workable clusters of analogous samples in the input data. This is commonly called *clustering*. This learning algorithm can provide suitable internal representation of the input information, by preprocessing the baseline input, so as to reposition it into a different variable space of the algorithm. The preprocessing stage can improve the outcome of a successive ML algorithm. This is typically referred to as a *feature extraction* [7].
3. Reinforcement learning. This learning algorithm deals with the problem of learning the appropriate action or sequence of actions to be taken for a given situation in order to maximize payoff [8].

This study will focus on supervised learning, since it is considered the main pillar of the IoT smart data analysis [5]. In a hot research topic in Information and Communications Technology (ICT) like ML algorithms for smart data analysis in IoT environment, there are many developments that quickly come into the spotlight and need to be highlighted in order to provide clear insights for researchers to choose the appropriate solution for a best performance. This article has incorporated information about supervised learning that are used in ML algorithms for smart data analysis in IoT environment, to achieve a precise, concrete and concise conclusion at the end of the study.

The key contributions of this study are presenting a comprehensive analyzes of the related literature on supervised ML techniques that is the main pillar of the IoT smart data analysis. This algorithm is investigated based on their respective sub-domains as well as advantages and limitations to achieving a precise, concrete and concise conclusion. This article also addresses research trends in the field of smart data analysis of IoT, and open issues that are being pursued in this area.

The rest of this paper is organized as follows. Section 2 discusses Taxonomies of supervised ML techniques based on their respective sub-domains with their advantages and limitations. Section 3 reviews the research trends and open issues. Section 4 elaborates the conclusions and recommendations.

## 2. TAXONOMIES OF SUPERVISED ML ALGORITHMS

The majority of practical ML uses supervised learning. In supervised learning, the available datasets are called "*true*" datasets or "*correct*" datasets. The algorithm is "*trained*" by using these input datasets. This is referred to as: training data. During this process, the algorithm makes predictions about the input data and expands or contracts its evaluations using the "*ground truth*" as baseline, repeating the process until the algorithm achieves a level of accuracy that is determined acceptable [11]. An ML algorithm will, typically, adjust satisfy a cost function. A cost function quantifies the error between the "*ground truth*" and algorithm calculations. Minimizing the cost function, allows

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

for training the model to yield results that align to more precise values (*ground truth*). Minimizing cost function can be achieved with the utilization of a *gradient descent technique* [12]. Different *gradient descent techniques* such as stochastic gradient descent, momentum-based gradient descent, Nesterov accelerated gradient descent [13] have been applied to ML training paradigms. In an example where '*m*' represents the number of trainings, each training can be denoted in a pair, as follows: (*x*, *y*). In this example the *x* can signify the input data and *y* signifies the class label. The input data *x* represents an *n* dimensional, while each dimension links to a explicit feature or a specific variable. In this example, the ML algorithm is aligned with a specific sensor system embedded in the program to accommodate the IoT application. [14]. Supervised learning problems can be further grouped into classification and regression problems [12]. The summarized taxonomy of supervised ML algorithms is given in Figure 2. In addition, Table 1 provides a summarized comparison of the basis and notable attributes, as well as advantages and limitations for each algorithm of the sub-domains of a supervised ML. In the following subsections, a detailed discussion is presented.

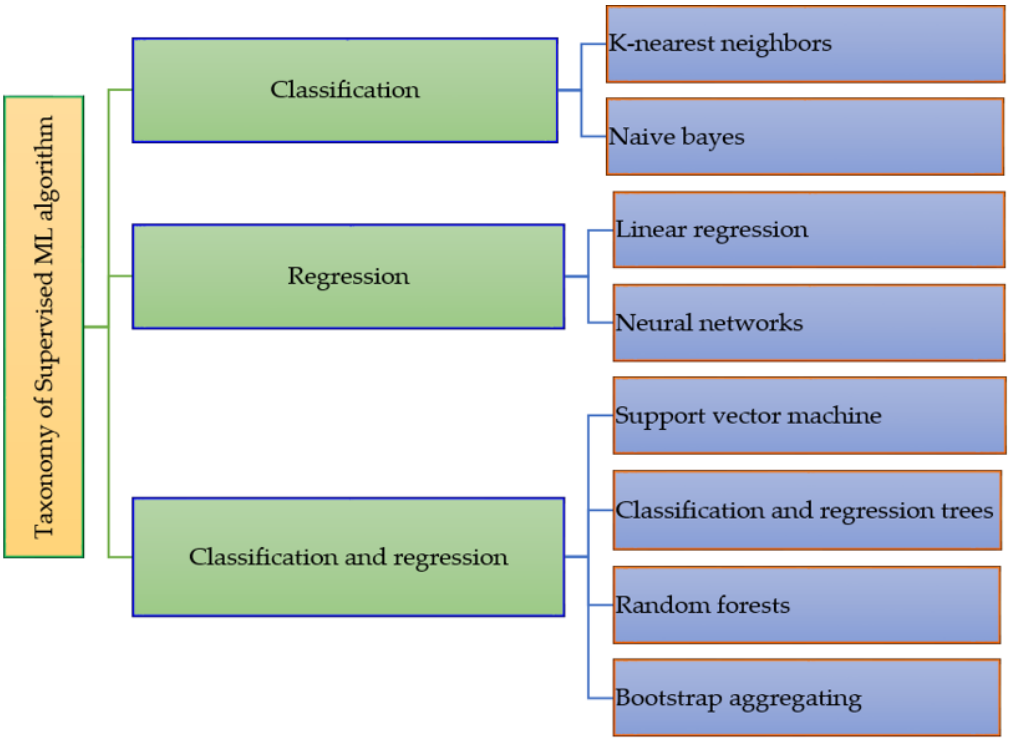


*Figure 2: Summarized taxonomy of supervised ML algorithms.*

## 2.1. Classification Tasks

Classification is a technique to categorize the data into a desired and distinct number of classes where a label is assigned to each class [15]. There are many methods to classify the data, a detailed discussion about the types of classification algorithms is given in the following subsections.

### 2.1.1. K-Nearest Neighbors (KNN)

The k-nearest neighbors (KNN) algorithm is a supervised ML algorithm that can be used to solve both classification and regression problems. However, it is more widely used in classification problems [16]. There are three important aspects that are used to evaluate any algorithm, (i) ease to interpret output, (ii) calculation time, and (iii) predictive power. KNN is simple, easy to implement, and commonly used because its ease of

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

interpretation and low calculation time. In classification and regression problems, the input consists of the $k$ that is closest to the training examples in the featured space. The output depends on whether KNN is used for classification or regression: (i) In KNN classification, the output is a class membership [5]. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its $k$ nearest neighbors ($k$ is a positive integer). If $k$ = 1, then the object is simply assigned to the class of its single nearest neighbor. (ii) In KNN regression, the output is the property value for the object. This value is the average of the values of $k$'s nearest neighbors. To find the $k$ of a data point, Euclidean distance, $L_\infty$ norm, angle, *Mahalanobis distance*, or *Hamming distance* can be used as the distance metric [17, 18]. A KNN model is shown in Figure 3, for $k$ =3, imagine that in this example, the test point (star) belonging to class B and for $k$=6, the point is classified as belonging to class A. In this example, KNN is a non-probabilistic and non-parametric model [19]. It is common for this to be the first choice for a classification study when no prior knowledge of the data distribution is available. In this illustration, KNN supplies all labelled input points. So, the question is raised what should be done with the unknown sample or samples? Resolving this dilemma can lead to significant computational expense. Classification of this type is based on a distance metric referred to as a similarity measure. Any sample labeled as unknown must be then classified by majority vote of its $k$ nearest neighbors. Because complexity increases as the dimensionality increases, dimensionality reduction techniques [20] must be performed before using KNN. This is necessary to circumvent effects that might eschew dimensionality. For example, KNN classifiers are used for stress detection in the monitoring of human physiological signals [21] as well as in the detection of seizure activity in a patient with epilepsy [22].

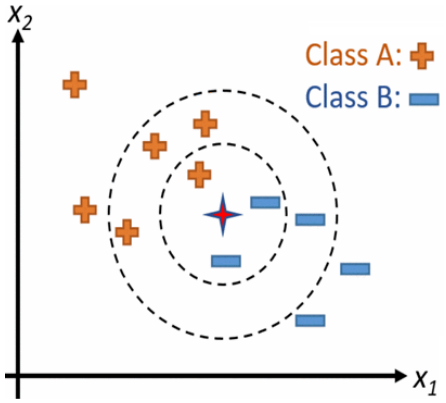


*Figure 3: A simple KNN model for different values of k.*

To formulate the problem in this example, Figure 3 represents the new input vector (data point) by $x$, its $K$ nearest neighbors by $N_k(x)$, the predicted class label for $x$ by $y$, and the class variable by a random variable $t$. In addition, 1(.) can indicate the function: 1($\boldsymbol{s}$)=1 if $s$ is true and 1($\boldsymbol{s}$)=0 otherwise. The form of the classification task can be expressed as follows [5]:

$$p(t = c|x, K) = \frac{1}{K} \sum_{i \in N_k(x)} 1(t_i = c) \qquad (1)$$

$$y = \max_c p(t = c|x, K)$$

Despite the benefits that can be achieved with this algorithm, (such as no training period) which allows for new data to be added seamlessly without negative impact on the accuracy of the algorithm; one major limitation of KNN is that it requires storing the entire training set, which makes KNN unsalable to large data sets. Moreover, the KNN algorithm doesn't work well with high dimensional data because with a large number of dimensions, it becomes difficult for the algorithm to calculate the distance in each dimension. In addition, the KNN is sensitive to noise in the dataset [23]. We need to manually input missing values and remove outliers. The authors in [24] have addressed with the large data sets issue by constructing a tree-based search with a one-off computation. Additionally, the authors in [25] suggest a structure for learning multiple metric combinations utilizing a vigorous and unique KNN classifier. Other authors, [26] link KNN with a *rough-set-based* algorithm that has been used for classifying travel pattern regularities.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

There are a lot of different improvements for the traditional KNN algorithm, such as the Wavelet Based KNN Partial Distance Search (WKPDS) algorithm [27], Equal-Average Equal-Norm Nearest Neighbor code word Search (EENNS) algorithm, and the Equal-Average Equal-Variance Equal-Norm Nearest Neighbor Search (EEENNS) algorithm [28].

### 2.1.2. Naive Bayes

A naive bayes classifier is a supervised machine-learning algorithm that uses the Bayes' Theorem, which assumes that features are statistically independent. The theorem relies on the naive assumption that input variables are independent from each other, i.e. there is no way to know anything about other variables when given an additional variable. Given a new, unseen data point (input vector) $x$= ($x_1$, …, $x_M$), naive Bayes classifiers, which are a family of probabilistic classifiers, classify $x$ based on applying Bayes' theorem with the "naive" assumption of independence between the features (attributes) of $x$ given the class variable $t$. By applying Bayes' theorem, the form can be expressed as follows [29]:

$$p(t=c|x_1,\dots,x_M)=\frac{p(x_1,\dots,x_M|t=c)p(t=c)}{p(x_1,\dots,x_M)} \quad (2)$$

By applying the naive independence assumption and some simplifications, the result is:.

$$p(t=c|x_1,\dots,x_M)\propto p(t=c)\prod_{j=1}^{M}p(x_j|t=c) \quad (3)$$

The form of the classification task can be expressed as follows [30]:

$$y=\max_c p(t=c)\prod_{j=1}^{M}p(x_j|t=c) \quad (4)$$

Where $y$ denotes the predicted class label for $x$. Different naive Bayes classifiers use different approaches and distributions to estimate $p(t=c)$ and $p(x_j|t=c)$.

Naive bayes classifier requires a small number of data points to be trained, can deal with high-dimensional data points, and is fast and highly scalable [31]. Moreover, Naive bayes classifier is a popular model for applications, such as spam filtering [32], text categorization, and automatic medical diagnosis [33]. On the other hand, the authors in [34] used this algorithm to combine factors to evaluate the trust value and calculate the final quantitative trust value of the agricultural product. Despite the benefits that can be achieved by this classifier; the main limitations of this classifier (Naive Bayes) are the assumption of independent predictors and assumption that all the attributes are mutually independent. However, in real life, it is almost impossible that we get a set of predictors which are completely independent [30]. Conversely, if the categorical variable has a category in the test data set, which was not observed in the training data set, then the model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as Zero Frequency. However, to solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation [35].

## 2.2. Regression Tasks

Regression models are used to predict a continuous value. A detailed explanation of the different types of regression tasks, with some important concepts are presented in following subsection.

### 2.2.1. Linear Regression

Linear Regression is a ML algorithm based on supervised learning. It performs a regression task. Regression models aim to provide a prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting possible results [36]. Regression models differ based on the relationship that exists between dependent and independent variables. The objective of linear regression is to learn a specific function $f(x, w)$. In this case one would plot the following: $f : \phi(x) \rightarrow y$. This is the linear amalgamation of a set of fixed linear or nonlinear functions from the input variable. This can be symbolized as the basic function: $\phi_i(x)$ [29].

$$f(x,w)=\phi(x)^{T}w \quad (5)$$

With $w$ signifying the weight vector (i.e., matrix), the equation would be conveyed as $w=(w_1, \dots, w_D)^T$, and $\phi = (\phi_1, \dots, \phi_D)^T$. A broad range of basic functions exist to assist in creating this application. For example: polynomial, gaussian, radial, or sigmoidal basic functions could be used in this application [37].

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A key concern is training the model for application. Several approaches are available: Ordinary Least Square, Regularized Least Squares, Least-Mean-Squares (LMS) and Bayesian Linear Regression. The LMS approach is very useful because it is quick, can easily be adapted to accommodate large data sets, and can learn the parameter requirements over the internet by using *stochastic gradient descent* (sequential gradient descent) [38]. Using the appropriate basic function, random nonlinearities in the mapping from input variable to output variable can be identified. However, the use of fixed basis functions can lead to significant shortcomings. (E.g. an increase in the dimension of the input space is coupled with rapid growth in the number of basic functions) [39]. Linear regression can process at a high rate [40]. For example, this algorithm can be used to analyze and predict the energy usage of buildings.

In contrast, neural networks can be used to fix the number of basic functions and allow the model to learn the parameters of the basic functions. In addition, neural networks are fast to process new data, because they are compact models. Additionally, they are easily adaptable to regression and classification problems. However, they usually require a large amount of computation to be trained. [41]. There exist many different types of neural networks, with different architectures, use cases, and applications.

### 2.3. Combining Classification and Regression Tasks

#### 2.3.1. Support Vector Machine (SVM)

Classical Support Vector Machine (SVM) is a support-vector network that can be utilized with supervised learning models. This model is a non-probabilistic, binary classifier that can be used to identify the hyperplane that divides classes of the training set. This provides a maximized margin. The predicted label of a previously unobserved data point can be determined by the side of the hyperplane on which it falls [42]. A significant property of SVMs is that it requires only a few training points. These training points are support vectors that can classify any new data point in the network. SVMs not only perform binary classification, they are also able to do multiclass classification. Four such models are: All-vs-all (AVA) SVM, One-vs-all (OVA) SVM, Structured SVM [43], and the Weston and Watkins version [44]. Besides linear classification, SVMs can perform non-linear classification. This can be useful for finding the hyperplane of a non-linear functioning input variable. For example, an input variable can be mapped into a high-dimensional feature space. This process is referred to as a *kernel trick* [45]. To formulate the problem, identify the typical vector of the hyperplane as $w$ and the parameter for controlling the offset of the hyperplane as $b$. To safeguard that SVM will be able to control for outliers in the data, a variable $\varepsilon_i$ can be introduced for every training point $x_i$. This is a *slack variable* that determines the distance that the training point encroached upon the margin in units of $|w|$. In this example, a binary linear classification task can be designated as a constrained optimization problem in the following manner [46]:

$$\min_{w,b,\varepsilon} f(w,b,\varepsilon) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \varepsilon_i \quad (6)$$

Subject to $y_i(w^T x_i + b) - 1 + \varepsilon_i \geq 0$

$i = 1, \ldots n$; $\varepsilon_i \geq 0$

Where parameter $C > 0$ determines how heavily a violation is punished. Moreover, the parameter $C$ is a hyperparameter which can be chosen via cross-validation or Bayesian optimization. There are different techniques to address the constrained optimization problem in Equation (6). P-pack SVM [49], quadratic programming optimization [47], and sequential minimal optimization [48] are techniques that can be applied to this problem. SVM is an excellent supervised learning model that can efficiently address high dimensional data sets. It is particularly effective addressing memory usage because it utilizes support vectors to facilitate prediction. However, this model has a significant drawback in that it does not directly provide probability estimates. SVM is useful in many real-world applications such as hand-written character recognition [50], image classification [51], and protein classification [52]. Finally, we should note that SVMs can be trained in an online fashion, which is addressed in [53]. The authors in [54] proposed a method on the Intel Lab Dataset; data set consists of four environmental variables (temperature, voltage, humidity and light) collected through *S4 Mica2Dot* sensors. The authors in [55] applied SVM to classify traffic data.

#### 2.3.2. Classification and Regression Trees (CART)

A training algorithm that works quite quickly is the classification and regression tree. This algorithm has been used to identify and classify

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*smart citizen* behaviors [56]. This decision tree algorithm is useful in predictive modeling machine learning. The classical decision tree algorithm, random forest, is an extremely useful procedure. For example, the input space in CART is first partitioned into axis-aligned cuboid regions $R_k$. Then a regression model assigns a separate classification to each region. The goal of this step is to predict the label for data points that may fall within a given region [57]. In the case of an unknown or hidden input vector (data point) $x$, the procedure for predicting the target marker can be determined based on the movement of the binary tree. This procedure must be consistent with the sequential decision-making process. An effective and efficient model for classification must predict a specific class for each region. The model must also be able to predict a constant for each region. To express the classification task, a class variable can be identified by a distinct random variable $t$. The predicted class label for $x$ by $y$ should also be able to be identified. The classification task would utilize the following formula [29],

$$p(t = c|k) = \frac{1}{|R_k|}\sum_{i \in R_k} 1(t_i = c)$$
$$y = \max_c p(t = c|x) = \max_c p(t = c|k) \quad (7)$$

Equation (7) states that it will be labeled by the most common mode in its corresponding region [29].

To formulate the regression task, we denote the value of the output vector by $t$ and the predicted output vector for $x$ by $y$. The regression task is expressed as,

$$y = \frac{1}{|R_k|}\sum_{i \in R_k} t_i \quad (8)$$

The output vector for $x$ will be the mean of the output vectors of data points in its corresponding region.

To train CART, the structure of the tree should be determined based on the training set. This means determining the split criterion at each node, along with the threshold parameter value. Finding the optimal tree structure is an NP-complete problem. This is termed as a greedy heuristic, which constructs the tree top-down and chooses the best split node by node to train CART. To achieve a better generalization and reduce overfilling, some stopping criteria should be used for constructing the tree. Possible stopping criterion are the maximum depth reached, whether the distribution in the branch is pure, whether the benefit of splitting is below a certain threshold, and whether the number of samples in each branch is below the criteria threshold. Moreover, after constructing the tree a pruning procedure can be used to reduce overfitting [56, 58]. The main advantage of CART is that it is fast and scalable to large data sets. However, it is very sensitive to the choice of the training set [59]. A significant shortcoming of this model concerns unsmooth labeling of the input space since each region of the input space is associated with exactly one label [9].

### 2.3.3. Random Forests

Random forests and random decision forests are useful learning methods for classification and regression tasks. These models construct multiple decision trees related to classification and regression; addressing training time, class output, mode of the classes, and mean prediction of individual trees. In random forests, multiple trees are trained in tandem. Each tree is trained with one subset of the training set. Each is chosen randomly with a potential replacement. This is done using a arbitrarily identified subset of $M$ input variables [60]. Two setups for forecasting the identifier of an unknown or unseen data point are suggested as follows: (1) A classification task would determine the mode of the labels predicted by each tree; (2) A regression tasks would identify the mean of the labels projected by each tree. However, an issue to address is the possibility of different values for $M$: A value for $M$ that is determined to be too small results in random trees with meagre predictive power. Similarly, value for $M$ that is too large results in random trees that are too similar to be useful for prediction.

Random forests have a high degree of accuracy. However, the cost of losing human interpretability is significant [61]. Random forests are fast and useful for large data sets. Random forests also have numerous real-world applications, for example body-pose recognition [62] as well as body-part classification.

### 2.3.4. Bootstrap Aggregating

Bootstrap aggregating (bagging), is an ensemble technique used to enhance accuracy and strength of ML algorithms. Bagging reduces the risk of overfitting. With this procedure, $K$ new $M$ sized training sets are randomly replacing data points from the original training set. In each newly created training set, a ML model is trained [63]. The predicted label of a different and unseen data point is set as the new mode of the labels predicted by

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

each model in the classification task and becomes the new mean in regression tasks. Of the different ML models, CART and neural networks bagging technique are very effect at refining the results. On the other hand, bagging degrades the performance of stable models such as KNN. It is, therefore, important to scrutinize its applicability to any specific task. Practical applications for bootstrap aggregating include customer attrition prediction and preimage learning [29].

*Table 1: Summarized comparison of the basis and notable attributes, as well as advantages and limitations of a supervised ML.*

| Data analysis tasks | ML algorithm | Advantages | Disadvantages |
|---|---|---|---|
| Classification | KNN | *Very simple implementation.<br>*New data can be added seamlessly.<br>*Robust against noisy training data.<br>*It has the capability to modeling complex classification problem by a collection of less complex local approximation.<br>*Maintain the information that presents in the training data. | *Does not work well with large dataset.<br>*Sensitive to unbalanced training data.<br>*It is supervised lazy learner.<br>Memory usage cost. |
| | Naive bayes | *Resulting interpretable model.<br>*Computational efficiency and highly scalable.<br>*Good classification performance.<br>*Require a small number of data points to be trained.<br>*It can deal with high-dimensional data points. | *It assumes that all the features are mutually independent. However, in real life, it is rarely that there is no correlation between features in raw data, which in turn leads to negatively on the classification accuracy. |
| Regression | Linear regression | *Model development is rapid and straightforward.<br>*Useful when the relationship to be modeled is not extremely complex and if don't have a lot of data. | *Applicable only if the solution is linear. In many real life scenarios, it may not be the case.<br>*Algorithm assumes the input residuals (error) to be normal distributed, but may not be satisfied always. |
| Combining classification and regression | SVM | *Regularization capabilities.<br>*Handles non-linear data efficiently.<br>*Solves both Classification and Regression problems.<br>*Stability.<br>*Provide better generalization capabilities. | *Choosing an appropriate Kernel function is difficult.<br>*Extensive memory requirement.<br>*Requires Feature Scaling.<br>*Time-consuming training.<br>*Difficult to interpret. |
| | Random forest | *It is considered one of the most robustness and accurate computational learning algorithms<br>*Good performance on many problem instances including non-linear.<br>*It has the capability of detect outliers and anomalies in knowledgeable data. | *Overfitting can easily occur.<br>*Need to determine the number of trees.<br>*Small perturbation in data can significantly modify the tree's structure, which in turn leads to produce inaccurate interpretations. |
| | Bootstrap aggregating | *They often provide better calcification accuracy results than those obtained by individual machine learning. | *Increasing of computational complexity.<br>*Loss of interaction among the individual networks during learning. |

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

## 3. RESEARCH TRENDS AND OPEN ISSUES

### 3.2. Privacy and Security

IoT consists of a vast number of different devices that are connected to each other and transmit huge amounts of data. IoT applications fall into different categories according to their unique attributions and features. Certain issues should be considering for running data analysis in IoT applications in an accurate manner. First, the privacy of collected data is highly sensitive issue, because data collection processes can include personal or business data. This privacy issue must be solved. Second, due to the vast number of resources available and simple-designed hardware in IoT, it is essential to consider security parameters, such as network security and data encryption. Ignoring security issues in the design and implementation of IoT devices can lead to an infected network.

### 3.2. Real-Time Data Analytics

According to the characteristic of smart data, analytic algorithms should be able to handle big data. That is, IoT requires algorithms that can analyze data that comes from a variety of sources in real-time. Many attempts have been made to address this issue. Deep learning algorithms, which are a form of neural networks incorporating evolution, can reach a high accuracy rate if they have enough data and time. However, deep learning algorithms can easily be influenced by noisy smart data. Furthermore, neural network-based algorithms can lack accurate interpretation. In the same manner, semi-supervised algorithms, which model a small amount of labeled data with a large amount of unlabeled data, can assist IoT data analysis.

## 4. CONCLUSION

This paper addresses the supervised ML techniques that are considered the main pillars of the IoT smart data analysis. To reach suitable decisions for smart data analysis, it is necessary to determine which task should be accomplished out of structure discovery, finding unusual data points, predicting values, predicting categories, or feature extraction. In order to predict values and classify sequenced data, the linear regression and SVM methods are the two most frequently applied algorithms. The objective of the models applied in these algorithms is to process and train data of high velocity. Another fast training algorithm is the classification and regression tree. To find unusual data points and anomalies in smart data, two important algorithms can be applied. Namely, the one-class SVM or PCA-based anomaly detection method. Both can train anomalies and noisy data with a high degree of accuracy. The SVM is a popular classification algorithm, which is capable of handling massive amounts of data and classifying their different types. Because SVM can handle a high volume and a variety of types of data, it is commonly applied in most smart data processing algorithms. To predict the categories of data, neural networks are suitable learning models for function approximation problems. Moreover, because smart data should be accurate and requires a long training time, a multi-class neural network can provide an appropriate solution. Research on the ML indicate that several challenges remain. This article has highlighted some shortcomings and challenges that exist with respect to some aspects of supervised ML techniques and future research that may prove beneficial in pursuing this vision as a useful technology.



**Conflicts of Interest:** The authors declare no conflict of interest.

## REFERENCES

[1] 1. Sharakhina, L.V. and V. Skvortsova. *Big Data, Smart Data in Effective Communication Strategies Development*. in *2019 Communication Strategies in Digital Society Workshop (ComSDS)*. 2019. IEEE.

[2] 2. Al-Fuqaha, A., et al., *Internet of things: A survey on enabling technologies, protocols, and applications.* IEEE Communications Surveys & Tutorials, 2015. **17**(4): p. 2347-2376.

[3] 3. Alzubi, J., A. Nayyar, and A. Kumar. *Machine learning from theory to algorithms: an overview*. in *Journal of Physics: Conference Series*. 2018. IOP Publishing.

[4] 4. Jagannath, J., et al., *Machine learning for wireless communications in the Internet of things: a comprehensive survey.* Ad Hoc Networks, 2019: p. 101913.

[5] 5. Kashyap, R., *Machine Learning for Internet of Things*, in *Next-Generation Wireless Networks Meet Advanced Machine Learning Applications*. 2019, IGI Global. p. 57-83.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[6] 6. Masegosa, A.R., et al., *AMIDST: A Java toolbox for scalable probabilistic machine learning.* Knowledge-Based Systems, 2019. **163**: p. 595-597.
[7] 7. Buskirk, T.D., et al., *An introduction to machine learning methods for survey researchers.* Survey Practice, 2018. **11**(1): p. 2718.
[8] 8. Luong, N.C., et al., *Applications of deep reinforcement learning in communications and networking: A survey.* IEEE Communications Surveys & Tutorials, 2019.
[9] 9. Schrider, D.R. and A.D. Kern, *Supervised machine learning for population genetics: a new paradigm.* Trends in Genetics, 2018. **34**(4): p. 301-312.
[10] 10. Lee, J.H., J. Shin, and M.J. Realff, *Machine learning: Overview of the recent progresses and implications for the process systems engineering field.* Computers & Chemical Engineering, 2018. **114**: p. 111-121.
[11] 11. Singh, A., N. Thakur, and A. Sharma. *A review of supervised machine learning algorithms*. in *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. 2016. IEEE.
[12] 12. Osisanwo, F., et al., *Supervised machine learning algorithms: classification and comparison.* International Journal of Computer Trends and Technology (IJCTT), 2017. **48**(3): p. 128-138.
[13] 13. Qu, G. and N. Li. *Accelerated distributed nesterov gradient descent for smooth and strongly convex functions*. in *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. 2016. IEEE.
[14] 14. Lee, J., et al. *Integrating machine learning in embedded sensor systems for Internet-of-Things applications*. in *2016 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*. 2016. IEEE.
[15] 15. Kanj, S., et al., *Editing training data for multi-label classification with the k-nearest neighbor rule.* Pattern Analysis and Applications, 2016. **19**(1): p. 145-161.
[16] 16. Maillo, J., et al., *kNN-IS: An Iterative Spark-based design of the k-Nearest Neighbors classifier for big data.* Knowledge-Based Systems, 2017. **117**: p. 3-15.
[17] 17. Chomboon, K., et al. *An empirical study of distance metrics for k-nearest neighbor algorithm*. in *Proceedings of the 3rd international conference on industrial application engineering*. 2015.
[18] 18. Prasath, V., et al., *Distance and Similarity Measures Effect on the Performance of K-Nearest Neighbor Classifier--A Review.* arXiv preprint arXiv:1708.04321, 2017.
[19] 19. Berisha, V., et al., *Empirically estimable classification bounds based on a nonparametric divergence measure.* IEEE Transactions on Signal Processing, 2015. **64**(3): p. 580-591.
[20] 20. Azar, A.T. and A.E. Hassanien, *Dimensionality reduction of medical big data using neural-fuzzy classifier.* Soft computing, 2015. **19**(4): p. 1115-1127.
[21] 21. Ghaderi, A., J. Frounchi, and A. Farnam. *Machine learning-based signal processing using physiological signals for stress detection*. in *2015 22nd Iranian Conference on Biomedical Engineering (ICBME)*. 2015. IEEE.
[22] 22. Sharmila, A. and P. Geethanjali, *DWT based detection of epileptic seizure from EEG signals using naive Bayes and k-NN classifiers.* Ieee Access, 2016. **4**: p. 7716-7727.
[23] 23. Garcia, L.P., A.C. de Carvalho, and A.C. Lorena, *Effect of label noise in the complexity of classification problems.* Neurocomputing, 2015. **160**: p. 108-119.
[24] 24. Lu, W., et al., *Efficiently Supporting Edit Distance Based String Similarity Search Using B $^+ $-Trees.* IEEE Transactions on Knowledge and Data Engineering, 2014. **26**(12): p. 2983-2996.
[25] 25. Do, C.-T., et al. *Multiple Metric Learning for large margin kNN Classification of time series*. in *2015 23rd European Signal Processing Conference (EUSIPCO)*. 2015. IEEE.
[26] 26. Ma, X., et al., *Mining smart card data for transit riders' travel patterns.* Transportation Research Part C: Emerging Technologies, 2013. **36**: p. 1-12.
[27] 27. Wang, H., et al., *Discriminative feature extraction via multivariate linear regression for SSVEP-based BCI.* IEEE Transactions on Neural Systems and Rehabilitation Engineering, 2016. **24**(5): p. 532-541.
[28] 28. Hilbe, J.M., *Practical guide to logistic regression*. 2016: Chapman and Hall/CRC.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[29] 29. Mahdavinejad, M.S., et al., *Machine learning for Internet of Things data analysis: A survey.* Digital Communications and Networks, 2018. **4**(3): p. 161-175.

[30] 30. Jadhav, S.D. and H. Channe, *Comparative study of K-NN, naive Bayes and decision tree classification techniques.* International Journal of Science and Research (IJSR), 2016. **5**(1): p. 1842-1845.

[31] 31. Xu, S., *Bayesian Naïve Bayes classifiers to text classification.* Journal of Information Science, 2018. **44**(1): p. 48-59.

[32] 32. Singh, G., et al. *Comparison between Multinomial and Bernoulli Naïve Bayes for Text Classification*. in *2019 International Conference on Automation, Computational and Technology Management (ICACTM)*. 2019. IEEE.

[33] 33. Pham, B.T., et al., *Spatial prediction of rainfall-induced landslides using aggregating one-dependence estimators classifier.* Journal of the Indian Society of Remote Sensing, 2018. **46**(9): p. 1457-1470.

[34] 34. Han, W., et al. *Data driven quantitative trust model for the internet of agricultural things*. in *2014 International Conference on the Internet of Things (IOT)*. 2014. IEEE.

[35] 35. Cherian, V. and M. Bindu, *Heart disease prediction using Naive Bayes algorithm and Laplace Smoothing technique.* International Journal of Computer Science Trends and Technology (IJCST), 2017. **5**(2).

[36] 36. Weichenthal, S., et al., *A land use regression model for ambient ultrafine particles in Montreal, Canada: A comparison of linear regression and a machine learning approach.* Environmental research, 2016. **146**: p. 65-72.

[37] 37. Hoffmann, J.P. and K. Shafer, *Linear regression analysis*. 2015: Washington, DC: NASW Press.

[38] 38. Montgomery, D.C., E.A. Peck, and G.G. Vining, *Introduction to linear regression analysis*. Vol. 821. 2012: John Wiley & Sons.

[39] 39. Robert, C., *Machine learning, a probabilistic perspective*. 2014, Taylor & Francis.

[40] 40. Derguech, W., E. Bruke, and E. Curry. *An autonomic approach to real-time predictive analytics using open data and internet of things*. in *2014 IEEE 11th Intl Conf on Ubiquitous Intelligence and Computing and 2014 IEEE 11th Intl Conf on Autonomic and Trusted Computing and 2014 IEEE 14th Intl Conf on Scalable Computing and Communications and Its Associated Workshops*. 2014. IEEE.

[41] 41. Glorot, X. and Y. Bengio. *Understanding the difficulty of training deep feedforward neural networks*. in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 2010.

[42] 42. Shifei, D., Q. Bingjuan, and T. Hongyan, *An overview on theory and algorithm of support vector machines.* Journal of University of Electronic Science and Technology of China, 2011. **40**(1): p. 2-10.

[43] 43. Nikam, S.S., *A comparative study of classification techniques in data mining algorithms.* Oriental journal of computer science & technology, 2015. **8**(1): p. 13-19.

[44] 44. Alber, M., et al., *Distributed optimization of multi-class SVMs.* PloS one, 2017. **12**(6): p. e0178161.

[45] 45. Ponte, P. and R.G. Melko, *Kernel methods for interpretable machine learning of order parameters.* Physical Review B, 2017. **96**(20): p. 205146.

[46] 46. Utkin, L.V., A.I. Chekh, and Y.A. Zhuk, *Binary classification SVM-based algorithms with interval-valued training data using triangular and Epanechnikov kernels.* Neural Networks, 2016. **80**: p. 53-66.

[47] 47. Lee, C.-P. and D. Roth. *Distributed box-constrained quadratic optimization for dual linear SVM*. in *International Conference on Machine Learning*. 2015.

[48] 48. Huang, X., L. Shi, and J.A. Suykens, *Sequential minimal optimization for SVM with pinball loss.* Neurocomputing, 2015. **149**: p. 1596-1603.

[49] 49. Díaz-Morales, R. and Á. Navia-Vázquez, *Distributed Nonlinear Semiparametric Support Vector Machine for Big Data Applications on Spark Frameworks.* IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2018.

[50] 50. Azim, R., W. Rahman, and M.F. Karim, *Bangla Hand-Written Character Recognition Using Support Vector Machine.* International Journal of Engineering Works, 2016. **3**(6): p. 36-46.

[51] 51. Liu, P., et al., *SVM or deep learning? A comparative study on remote sensing image classification.* Soft Computing, 2017. **21**(23): p. 7053-7065.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[52] 52. Cang, Z., et al., *A topological approach for protein classification.* Computational and Mathematical Biophysics, 2015. **3**(1).
[53] 53. Wahab, O.A., et al., *CEAP: SVM-based intelligent detection model for clustered vehicular ad hoc networks.* Expert Systems with Applications, 2016. **50**: p. 40-54.
[54] 54. Khan, M.A., et al. *A novel learning method to classify data streams in the internet of things*. in *2014 National Software Engineering Conference*. 2014. IEEE.
[55] 55. Nikravesh, A.Y., et al. *Mobile network traffic prediction using MLP, MLPWD, and SVM*. in *2016 IEEE International Congress on Big Data (BigData Congress)*. 2016. IEEE.
[56] 56. Breiman, L., *Classification and regression trees*. 2017: Routledge.
[57] 57. Krzywinski, M. and N. Altman, *Points of Significance: Classification and regression trees*. 2017, Nature Publishing Group.
[58] 58. Wuest, T., et al., *Machine learning in manufacturing: advantages, challenges, and applications.* Production & Manufacturing Research, 2016. **4**(1): p. 23-45.
[59] 59. Hagenauer, J. and M. Helbich, *A comparative study of machine learning classifiers for modeling travel mode choice.* Expert Systems with Applications, 2017. **78**: p. 273-282.
[60] 60. Belgiu, M. and L. Drăguţ, *Random forest in remote sensing: A review of applications and future directions.* ISPRS Journal of Photogrammetry and Remote Sensing, 2016. **114**: p. 24-31.
[61] 61. Biau, G. and E. Scornet, *A random forest guided tour.* Test, 2016. **25**(2): p. 197-227.
[62] 62. Selvi, S.T., et al. *Text categorization using Rocchio algorithm and random forest algorithm*. in *2016 Eighth International Conference on Advanced Computing (ICoAC)*. 2017. IEEE.
[63] 63. Hassan, A.R. and M.I.H. Bhuiyan, *Computer-aided sleep staging using complete ensemble empirical mode decomposition with adaptive noise and bootstrap aggregating.* Biomedical Signal Processing and Control, 2016. **24**: p. 1-10.