Proceedings of the 2nd International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/81/20241144

Implementation of Distributed Machine Learning in Cloud Calculation: Evidence from Consumer Behavior Prediction

Runze Ji

School of Automation, Beijing Institute of Technology, Beijing, China

1320211079@bit.edu.cn

Abstract. In the field of machine learning, distributed machine learning (DML) has gained massive popularity in research and appeared in a wide range of applications including transportation, medicine, and business. Distributed machine learning showed huge potential in processing large amounts of data while keeping the data private. In this paper, a distributed implementation of a deep neural network (DNN) with a large dataset from a telecom company, aiming to predict consumer behavior based on usage data is carried out. Various datasets with distinct heterogeneity settings were applied to the network, simulating a real-world cloud application scenario. Comparisons between both the centralized model and the distributed models were made to analyze the impact on model performance with device heterogeneity and data heterogeneity. These result indicate that under certain circumstances, DML models showed better accuracy and lower loss compared to the centralized model. The paper provided some insights and made some assumptions on how the data and the parameters affect the model performance.

Keywords: Distributed machine learning, deep neural network, consumer behavior analysis, cloud computing.

1. Introduction

Machine learning (ML) has gained a respectable position in commercial and consumer behavioral analysis in the past years. Traditionally, early-stage consumer behavior analysis and prediction relied on methods like surveys, questionnaires, and experimental design. With the increase in data size, gaining complexity of research topics, and more importantly, the advances in computational power, machine learning has been gradually introduced into the field. The ability to understand and learn patterns from high-dimension and large data, makes machine learning methods appealing to researchers and companies. Nowadays, online shopping and the extensive use of social media platforms like Facebook, Instagram, and YouTube make purchasing products easier than ever, there are already ways of analyzing big data pulled from these platforms and being used to predict consumer behavior and enhance online product promotion [1]. User-generated content like comments, reviews, and ratings are vital for companies to get insights into the customers' demands and intention of purchasing, K-nearest neighbors (KNN), random forest classifier, Bayesian classifier, and many other classifiers were used to help e-commerce organizations accurately apply marketing on target consumers [2]. Another study combines novel approaches with unsupervised clustering algorithms to reveal how various aspects of social network marketing affect consumer purchase behavior [3]. Time spent reading product info on e-

DOI: 10.54254/2755-2721/81/20241144

commerce websites can also have an influence [4]. Not only do businesses benefit from such studies, but with smart home accessories, machine learning algorithms can detect patterns and user preferences of inhabitants to save energy while retaining the same level of comfort [5].

Exponential growth in data has driven the application of distributed machine learning, making it the center of attention, combined with the huge demand for computational power, and the increasing privacy concerns. The use of distributed machine learning (DML) aims to solve these problems by scaling across multiple platforms, while preserving privacy, making it a good choice for various domains including the Internet of Things (IoT), autonomous driving vehicles, marketing, and so on. Federated learning (FL), being one of the most studied areas of DML, trains models with other devices in the network without sharing local data, so privacy is preserved [6]. In this case of studying and predicting consumer behavior, companies can train models locally and don't worry about compromising consumer privacy. When dealing with distributed machine learning, the challenges that come with it are communication and scalability. Recent studies focused on optimization techniques that can reduce communication overhead, and result in faster model convergence [7]. Differential privacy and secure multiparty computation further ensure data privacy for federated learning [8], Another major concern is the device heterogeneous issue, some devices have powerful hardware that can do multiple calculations in a cycle, while low-powered devices like IoT devices have performance constraints. Researchers have created a neural network framework that reduces the communication overhead to make FL available on these devices [9]. Certain applications can only utilize distributed machine learning, which has large amounts of data and requires computing power, like terabytes of enterprise data, and astronomical data that are impossible to move and centralize [10]. Although distributed machine learning has great capabilities like privacy-preserving features and scalability, however, device heterogeneity, communication overhead, and more advanced privacy protection methods still require improvement.

The motivation of this paper is mainly to conduct an experiment that simulates real-world usage of distributed machine learning. This study selected a dataset from a telecommunications company that includes its customer data, the goal is to train a distributed model that ensures privacy, like in real-world scenarios where the various branches of the same company do not share its customer data whether due to privacy concerns or competition. This paper simulates that scenario and discusses the pros and cons of going distributed. This study will first cover the data specifications and methods used in this study, then, this research will briefly look at the algorithms and model structure, and parameter settings, important assessment metrics will be introduced as well, Finally, it will dive into the phenomenon and make comparisons on differences between centralized machine learning and distributed machine learning with various heterogeneity setting.

2. Data and method

This paper worked on a dataset with a size of 320,000 lines, 25 input features, and 9 output class labels. The dataset was from a telecom company, used to analyze the mobile phone usage habits of different types of users based on their behaviors, thus recommending suitable cellular plans for the customer. This study intends to perform the model training on 2 worker nodes, each worker node was assigned 150,000 lines of selected data, and the remaining 20,000 was used as the validation set. Input features like the current service type, the total fee in the last 4 months, and the cellular data usage were taken into account to determine the best possible cellular plan for the user. For the specifications of the dataset, this research applied 2 types of data heterogeneity settings on the dataset, one is of moderate level of heterogeneity, the other is of strong level. By the term of "moderate" heterogeneity, the dataset has 9 class labels, each of the 2 datasets has a distinct categorical bias, 1 dataset has a bias on the 4 out of 9 labels, the other has a bias on the other 5 out of a total of 9 labels. Still, they share the same 9 label categories despite the size difference. For the term "strong" heterogeneity, the 2 datasets each have their distinct class labels, meaning that they do not overlap, simulating a scenario of completely heterogeneous data across worker nodes used in real-world applications.

For this particular case, a deep neural network (DNN) was used to conduct this study. I used DNN primarily due to its ability to capture non-linear relationships. The deep neural network used in this study

DOI: 10.54254/2755-2721/81/20241144

consists of an input layer that has 25 neurons, and an output layer of 9 neurons which is the total number of predicted labels. Three hidden layers with 192, 96, and 48 respectively. All layers are fully connected, and ReLU activation functions were used in all layers. This study set the L2 regularization coefficient to 10-5 to prevent model overfitting. Speaking of the training procedure, the input 25 class labels were directly fed into the network, and the class labels were encoded with a one-hot encoder. Cross-entropy loss was used to measure the performance of the model during classification. Momentum stochastic gradient descent (Momentum SGD) optimizer with a learning rate of 0.001, momentum of 0.9, batch size of 128, L2-regularization coefficient of 10^{-5}, and dampening of 0 is applied. Based on trials and observations, all networks were set to train 10 global rounds, with the 2 local round parameters being the only variables. The neural network is built with PyTorch, a Python framework for machine learning and deep learning, combined with a federated machine learning framework called 'flower' which utilizes gRPC to communicate between nodes and the parameter server. The performance of the trained models was measured by validation accuracy and validation loss. A centralized ML model is also trained as the control group.

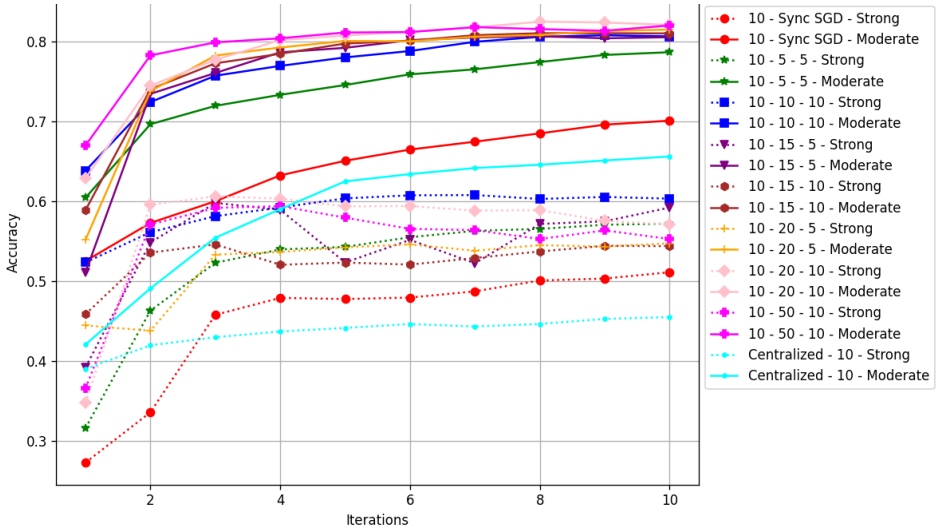

Figure 1. Iteration–Accuracy graph of all model configurations (Photo/Picture credit: Original).

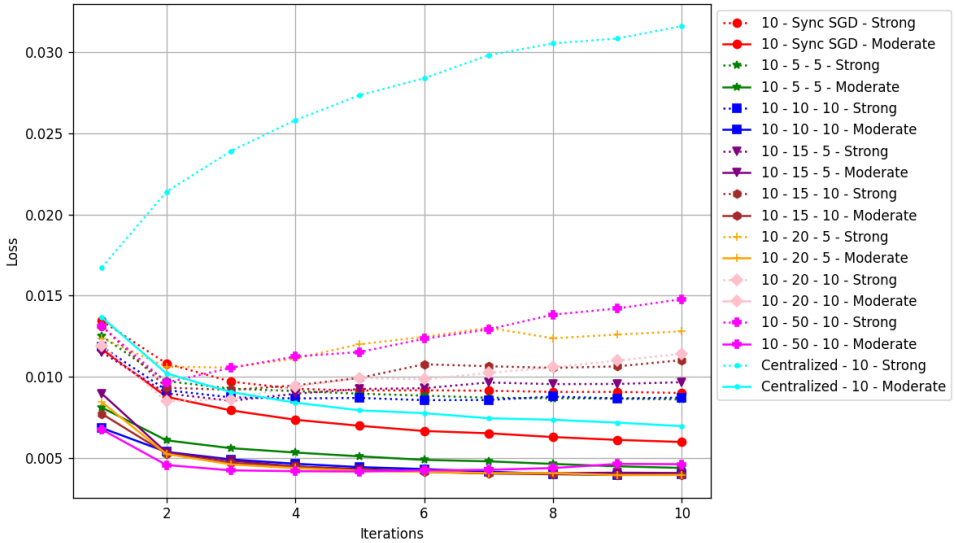

Figure 2. Iteration–Loss graph of all model configurations (Photo/Picture credit: Original).

DOI: 10.54254/2755-2721/81/20241144

3. Results and discussion

3.1. Model Performance

Fig. 1 shows the iteration-accuracy graph and Fig. 2 shows the iteration-loss graph. All the worker node configurations used were shown in the results, and centralized control groups with cyan lines were added for comparison. The legend indicates the training parameters, e.g., '10-15-5-Strong' means 10 global rounds, 15 local rounds on worker node 1, and 5 local rounds on node 2. The centralized group scored an accuracy of 45.51% and 65.61% (moderate and strong heterogeneity), respectively. For the 'moderate heterogeneity' group, the best accuracy was 82.07%, achieved with '10-20-10-Moderate' (10 global rounds, 20 rounds on worker node 1, and 10 rounds on node 2). As for the 'strong heterogeneity' group, the best accuracy was 60.33%, achieved with '10-10-10-Strong' local rounds on both worker nodes.

3.2. Comparison and Explanation

It can be seen in Fig. 1 that the dataset with a moderate level of heterogeneity represented by solid lines has an improvement of around 20% over the strong heterogeneity ones with dotted lines. If one looks at the solid lines, except for the green solid line (10-5-5-Moderate) and the red solid line (10-Sync SGD-Moderate), all other models have very little difference in accuracy. There were also differences in losses, if one temporarily excluded 'Centralized-10-Strong', as shown in Fig. 3.

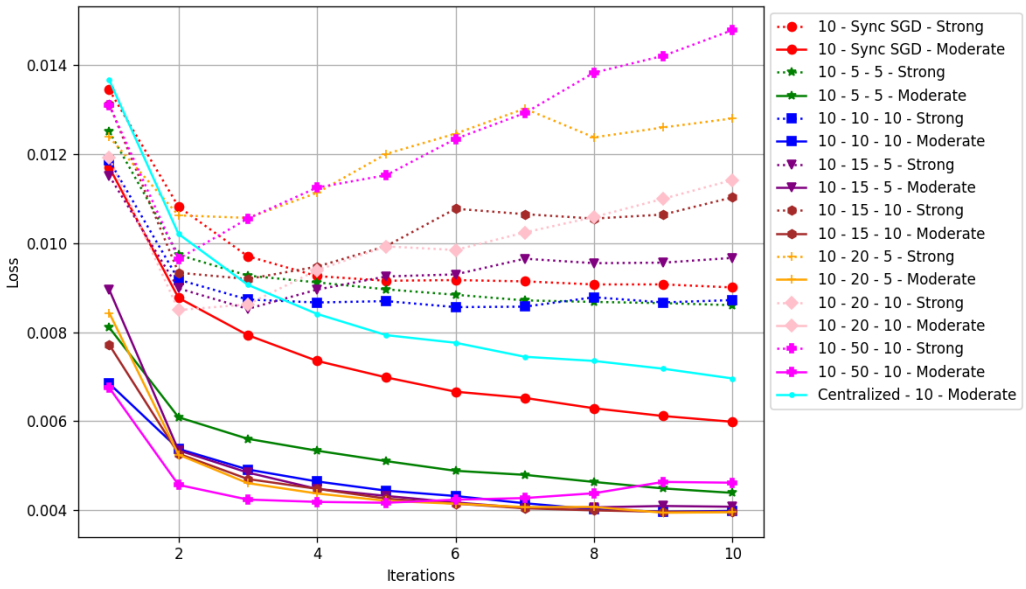

Figure 3. Iteration–Loss graph of all models without 'Centralized-10-Strong' (Photo/Picture credit: Original).

Seen from Fig. 3, the lines can be roughly divided into 2 groups, the upper group with the dotted lines are the models with strong heterogeneity. With a strong heterogeneity level, all have higher losses compared to their moderate heterogeneity counterparts. Seen in Fig.2 that models trained on strong heterogeneity data have higher losses. If one looks at the magenta, orange, pink, and brown dotted lines, the loss starts to rise as the number of iterations increases. Taking a further look, if one calculates the ratio between local round τ_1 and τ_2, the final loss increases as the ratio increases, indicating that the ratio between the final loss and the number of local rounds between the two worker nodes is positively related. In the case of moderate heterogeneity, all distributed models have higher accuracy and lower loss than the centralized one, one can say that under the circumstances of this specific study, distributed machine learning can yield better results than the centralized ones (seen from Table 1).

DOI: 10.54254/2755-2721/81/20241144

Table 1. Loss comparison between different worker rounds with strong heterogeneity

Global round	Worker 1 rounds τ_1	Worker 2 rounds τ_2	Ratio $\frac{\tau_1}{\tau_2}$	Final loss
10	15	10	1.5	0.0110
	20	10	2	0.0114
	20	5	4	0.0128
	50	10	5	0.0147

Table 2. Loss comparison between different worker rounds with moderate heterogeneity

Global round	Worker 1 rounds τ_1	Worker 2 rounds τ_2	Ratio $\frac{\tau_1}{\tau_2}$	Final accuracy
10	10	10	1	80.64%
	15	10	1.5	81.00%
	20	10	2	82.07%
	20	5	4	81.48%
	50	10	5	82.00%

There is an interesting result shown in Table 2, one can see that the accuracy peaked at 57.11% with 20 rounds at worker node 1 and 10 rounds at worker node 2. Usually with distributed models, accuracy tends to go down as local rounds τ increases, the local models diverge from each other towards their local minimum. If one deliberately makes worker 1 do more local rounds, the model accuracy saw a slight improvement, it is assuming that the global minimum of the objective function is closer to the local minimum of worker 1, meaning that the data held by worker 1 is more similar to the global data. So, in practical applications, balancing data between worker nodes is an issue that needs to be taken into consideration. A comparison between different local rounds is conducted. It is selected distributed models with the same 10 global rounds and the same amount of local rounds, local rounds were set to 10, 5, and 1 (Synchronous SGD) respectively. In this case, one can determine how local training rounds can affect model performance by analyzing the iterations-accuracy curve shown in Fig. 4.

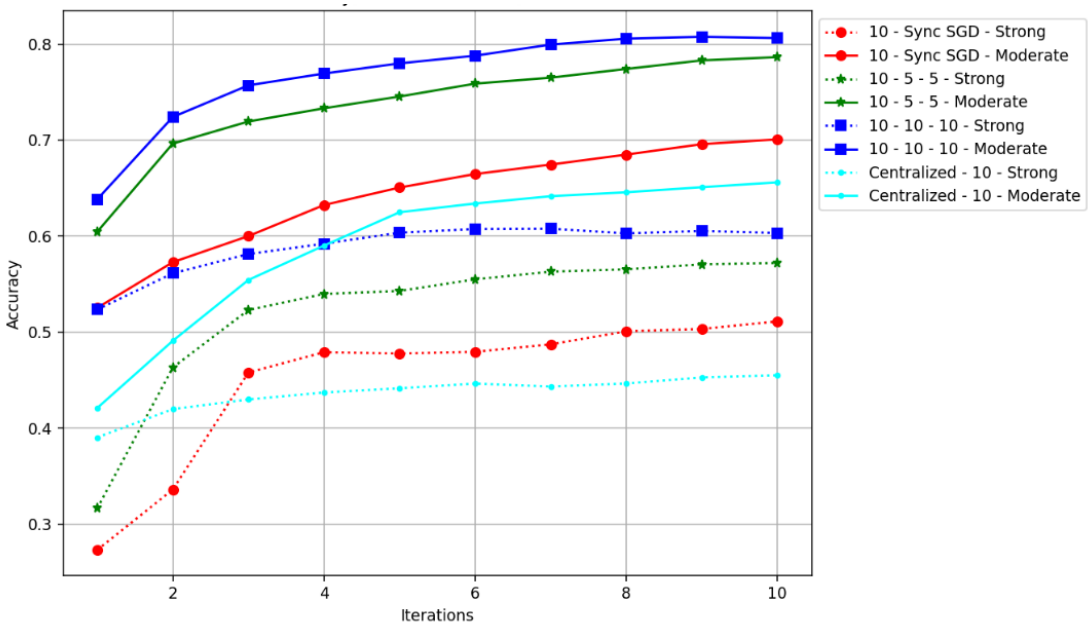

Figure 4. Iteration–Accuracy graph of models with the same local rounds (Photo/Picture credit: Original).

DOI: 10.54254/2755-2721/81/20241144

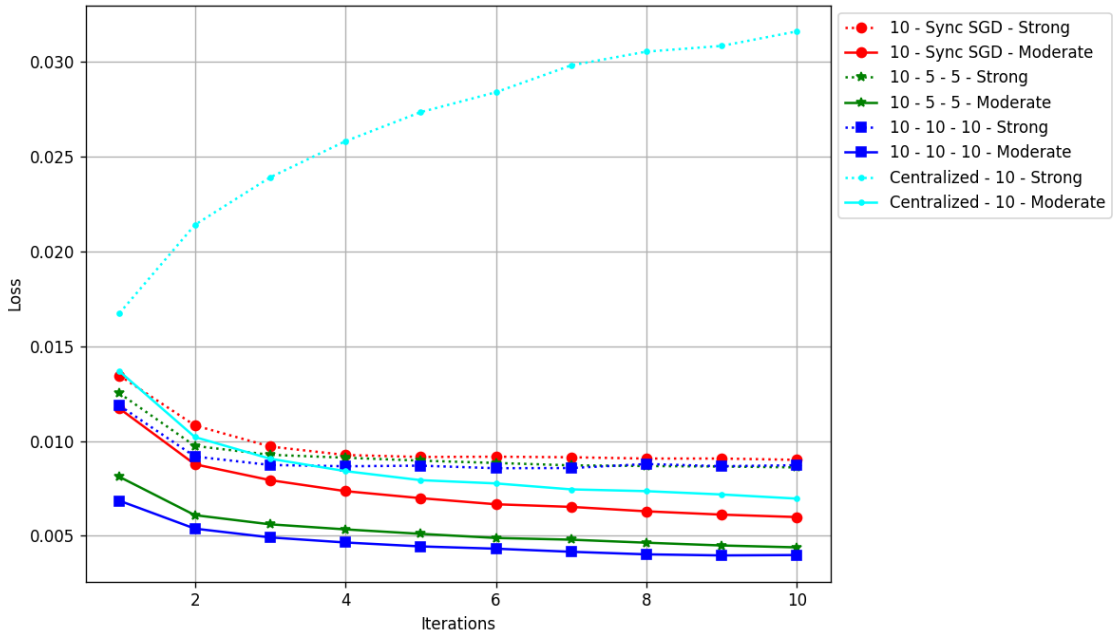

Figure 5. Iteration-Accuracy graph of models with the same local rounds (Photo/Picture credit: Original).

The blue lines with 10 local rounds perform the best in each of their categories, followed by green lines and red lines, and finally the cyan lines. Seen from Fig. 4 and Fig. 5, the higher the number of local rounds, the better the model's accuracy and the lower the model's loss. Looking at the lines respectively, one can see that with more local update rounds τ, the loss becomes lower and the accuracy becomes higher. Conventionally, for a Lipschitz smooth function, after t iterations of local update, SGD is bounded as:

$$\mathbb{E}\left[\frac{1}{t}\sum_{k=1}^{t}\|\nabla F(x_k)\|^2\right] \leq \frac{2\left[F(x_1)-F_{inf}\right]}{\eta t} + \frac{\eta L\sigma^2}{m} + \eta^2 L^2\sigma^2(\tau-1) \quad (1)$$

As local round τ increases, the last term increases, leading to a growing error floor. So theoretically, larger τ gives inferior error convergence, which is based on the assumption that the objective function is Lipschitz smooth. The probable cause for this unconventional conclusion is that the dataset does not meet the requirements for Lipschitz smoothness, so the conclusion "Larger τ gives inferior error convergence" may not hold in this particular case.

3.3. Limitation and prospects

In this paper, only DNN has been discussed as a cloud-distributed algorithm, However, traditional ML algorithms like XGBoost with FedXGBBagging, these algorithms are also suitable for this study, but due to the limitation of the label encoder, the dataset with strong heterogeneity setting, which has no shared labels between worker nodes cannot be encoded and it is challenging to aggregate distributed models. Furthermore, the number of worker nodes in the cloud is also limited to 2, as more worker nodes are added to the system, the result may vary depending on the configuration and the number of nodes. This limitation is the biggest restriction towards the scalability of the proposed method of this study.

Future work on this study can focus on several areas. Experiment with other distributed machine learning algorithms. This study only focused on neural networks specifically, other traditional machine learning algorithms like decision tree, random forest, and XGBoost are also great for such classification tasks. Although there are aggregation methods like Federated XGBoost Bagging (FedXGBBagging) Finding a proper method of model aggregation is still a problem to solve. Further investigation on the implications of Lipschitz smoothness assumption on the convergence analysis of local update SGD is

DOI: 10.54254/2755-2721/81/20241144

crucial, more study into the properties of the objective function and how it affects the model convergence would be valuable. While this paper covered the effects of different local rounds on model accuracy and loss, conducting more experiments with other hyperparameters, such as learning rate, batch size, or the number of global rounds, can help determine the optimal setting for various scenarios. Other techniques like performing grid searches and learning rate decay can be implemented to improve performance. These methods can help determine the optimal model structure and parameter setting. By addressing these areas, one can expect to build a more robust distributed machine learning system that can deliver high accuracy.

4. Conclusion

To sum up, the main objective of this paper is to explore the application of distributed machine learning in cloud calculations, specifically in consumer behavior prediction. The results show that distributed machine learning is able to handle large datasets and achieve good performance in consumer behavior prediction. It is found that using DNN as a distributed machine learning algorithm can obtain better results than the centralized. It is discovered that the performance tends to improve when dealing with strong heterogeneity data. Further research into the application of distributed machine learning in cloud computing and attempts to use other algorithms and techniques deserve deeper investigation. This research lies in its ability to provide insights and methods for the application of distributed machine learning in cloud computing, through the experiments in this paper, one discovers some of the potential of applying distributed machine learning in consumer behavior prediction, which has implications for both business and society.

References

[1] Chaudhary K, Alam M, Al-Rakhami M S and Gumaei A 2021 Machine learning-based mathematical modelling for prediction of social media consumer behavior using big data analytics Journal of Big Data vol 8(1) p 73

[2] Shrirame V, Sabade J, Soneta H and Vijayalakshmi M 2020 Consumer behavior analytics using machine learning algorithms IEEE International Conference on Electronics Computing and Communication Technologies (CONECCT) pp 1-6

[3] Ebrahimi P, Basirat M, Yousefi A, Nekmahmud M, Gholampour A and Fekete-Farkas M 2022 Social networks marketing and consumer purchase behavior: the combination of SEM and unsupervised machine learning approaches Big Data and Cognitive Computing vol 6(2) p 35

[4] Necula S C 2023 Exploring the impact of time spent reading product information on e-commerce websites: A machine learning approach to analyze consumer behavior Behavioral Sciences vol 13(6) p 439

[5] Schweizer D, Zehnder M, Wache H, Witschel H F, Zanatta D and Rodriguez M 2015 Using consumer behavior data to reduce energy consumption in smart homes: Applying machine learning to save energy without lowering comfort of inhabitants IEEE 14th International Conference on Machine Learning and Applications (ICMLA) pp 1123-1129

[6] Kairouz P, McMahan H B, Avent B, Bellet A, Bennis M, Bhagoji A N and Zhao S 2021 Advances and open problems in federated learning Foundations and trends® in machine learning vol 14(1–2) pp 1-210

[7] Li T, Sahu A K, Talwalkar A and Smith V 2020 Federated learning: Challenges methods and future directions IEEE signal processing magazine vol 37(3) pp 50-60

[8] Bonawitz K 2019 Towards federated learning at scale: System design arXiv preprint arXiv:190201046

[9] Gao Y, Kim M, Thapa C, Abuadbba A, Zhang Z, Camtepe S and Nepal S 2021 Evaluation and optimization of distributed machine learning techniques for internet of things IEEE Transactions on Computers vol 71(10) pp 2538-2552

[10] Verbraeken J, Wolting M, Katzy J, Kloppenburg J, Verbelen T and Rellermeyer J S 2020 A survey on distributed machine learning Acm computing surveys (csur) vol 53(2) pp 1-33