Intelligent Data Forwarding Scheme for LoRa based Fog Enabled Smart Agriculture

Mayank Vyas[1], Garv Anand[1], Sanjeet Kumar Nayak[2], Ramanarayan Yadav[1]

[1]IITRAM, Ahmedabad, India, [2]IIITDM Kancheepuram, Chennai, India

***Abstract*—Nowadays, Internet of Things (IoT) technologies are very useful to transform traditional farming practices into smart farming across the globe. Since most of the agricultural land is located in remote places, there is a need for new technologies for making them smart. For smart farming, low-cost, and resource-constraint sensors are placed across the agricultural field. To provide continuous connectivity for data communication, there must be a suitable and reliable transmission over long distances and, at the same time, should consume less energy. In particular, Long Range (LoRa), a relatively new communication technology, uses long waves to work over long distances. This is extremely useful in agriculture, where the communicating areas are broad fields of crops and Internet connectivity is low and/or intermittent. So, to reduce data transmission between layers (sensing, fog and cloud), a machine learning based data forwarding architecture is proposed for the three tier architecture, namely, Regressive Prediction Data Forwarding Model (RPDM). A suitable and lightweight machine learning model is deployed on each layer which will predict the next coming sensing data. So, the upcoming data will not be forwarded from the sensing layer to the fog layer if they match with the predicted one. Also, in the case of Internet failure, the sensing layer made capable of performing the actuation based on current data value. We validated the performance of RPDM on a real testbed in terms of the amount of energy consumed and data transmitted.**

***Index Terms*—Smart agriculture, LoRa, Data reduction, Machine learning, IoT, Fog/Edge computing**

I. Introduction

Agriculture has always been one of the most critical sectors across the globe. With the growing interest and demand for intelligent agriculture, there is a need to convert traditional agricultural land to smart agricultural land. For smart farming, low-cost, and resource-constraint sensors are placed across the agricultural field to collect various soil and crop parameters. These sensors are not capable enough to process the data, thus, need to communicate to large storage servers like cloud. To provide continuous connectivity there must be a suitable and reliable technology for data communication over long distances and, at the same time, should consume less energy.

At present, cloud-based infrastructures [1] are being utilized to support various smart agriculture applications, wherein the data from smart sensors in the agricultural field is transmitted to the cloud over the Internet. Though cloud-based infrastructures certainly offer enormous processing power and storage capacity, there are two key limitations that need to be addressed when used in the context of smart agriculture: (i) sensor data transmitted over the Internet requires continuous Internet connectivity, (ii) IoT devices must transmit data continuously to the cloud for storage, thus, the energy of battery-powered IoT devices is quickly drained. These two limitations of cloud-based infrastructure for smart agriculture can be overcome with the help of fog computing and Machine Learning. To address these limitations, we propose a LoRa-enabled fog based smart agriculture infrastructure using and Machine Learning (ML). This will reduce the quantity of data transferred from sensors to fog and cloud. Our proposed framework consists of three layers (i) sensing layer, (ii) fog layer, and (iii) cloud [2]. Our aim is to reduce the number of data transmissions from the sensing layer to the fog layer within an accuracy threshold (called ε). This will help in increasing the lifetime of deployed sensors in the field. To achieve the objective, we deploy the same ML model at the sensor nodes as well as the fog node. A sensor node transmits the sensed data to the fog node only if the error (difference between predicted and actual value) is beyond the pre-defined threshold (ε). Since the agricultural fields are slow changing dynamics therefore we won't be getting very deviated readings. Another major problem which we addressed is that the size of the ML model is huge and deploying it at the edge devices (sensor nodes) will consume more energy. So, the trained ML model is converted into a lightweight flat buffer file format to address the issue using TensorFlow and then uploaded to the edge devices. The same machine learning model runs on both edge and fog layers and the model will continuously predict the sensor data. This removes the frequency of data transmissions between the fog layer and edge layer since there is no need to send the data if the data being predicted on both layers is in the permissible error range $(x[n] \pm \varepsilon)$, where $x[n]$ is the actual sensor value and ε is error threshold.

As shown in Figure 1, this scheme also resolves the issue of data breaches in the transmission process. Since the machine learning model is a heavy model it is quite difficult to deploy over sensor nodes with low flash memory; which will increase their power consumption. So, a trained model is converted to a lightweight flat buffer file and then we deploy it onto the sensor node. The only few cases when the data will be sent from field to fog are if there is a variation of readings from the set threshold. In such cases, the machine learning model running at the sensing layer will adapt and change in the predictor's variables state, and the newly updated weights are transmitted to the higher level that is fog level via LoRa Protocol. After that,

the sensor node will send the updated predictor's variable state to the fog node so that the fog node can also adapt to changes. This ensures that at both layers of sensing and fog, the same ML model will be running. Also, in case of intermittent Internet connectivity at the fog layer to the edge layer, the same model is compressed in TensorFlow Lite file which is compatible with performing actuations like turning on the water sprinkle in case of low moisture level in the field.

The outline of this paper is as follows. Section II presents the related works. Section III presents the proposed ML architecture. In Section IV, performance of the proposed scheme is analysed. Finally, Section V concludes the paper.

II. Related Works

With the proliferation of certain research works, the use of IoT and machine learning has been proven to be vital in domains like system automation and reducing dependencies on the Internet. Many research works have been published and are under research to levitate the performance of the architecture by taking the crust. In [3], [4], a Ubiquitous IoT framework has been designed to obviate human intervention in a system along with the IoT framework. A machine learning model has been designed to provoke security into the framework. This architecture just focuses on security threats but in our model we not only have reduced the security threats but also have taken into consideration the bandwidth issue. As discussed in [5], [6], some machine learning Model along with their application in Wireless Sensor Networks (WSN's) has been discussed and analysed. Also an Idea of reinforcement Learning has been highlighted in this research. In [7], the authors have discussed about the advanced reinforcement learning application in smart farming and also they have used Internet of things to curb the dependency on Internet. The application works for four layer architecture. They focused on improving the crop yield by monitoring the water availability in the soil. In [8], the authors have discussed about the data reduction architecture from fog to cloud using ADE algorithm and also they had suggested a FSP structure for placing the sensors on the field. The article also cites that the reduction in delay is around 5.6% with resource usage upto 13.2%.

III. Proposed Machine Learning Architecture

In this section, we discuss the ML model used for predictions. We use supervised machine learning as the data labels like temperature, humidity, soil moisture, etc., are already known. The prediction which is the output of the model is being sampled on one of these labels. Supervised learning is widely used for two purposes **Classification** and **Regression**. In this paper, the Regression approach is used since the model is to be trained to predict the data rather than classifying them. The flowchat of the proposed model is shown in Figure 1.

The aim of this work is to minimize the data transmission cost and to deploy a machine learning model to predict those values that are more likely to be given by the sensors in the sensing layer. The same ML model will run on the edge layer

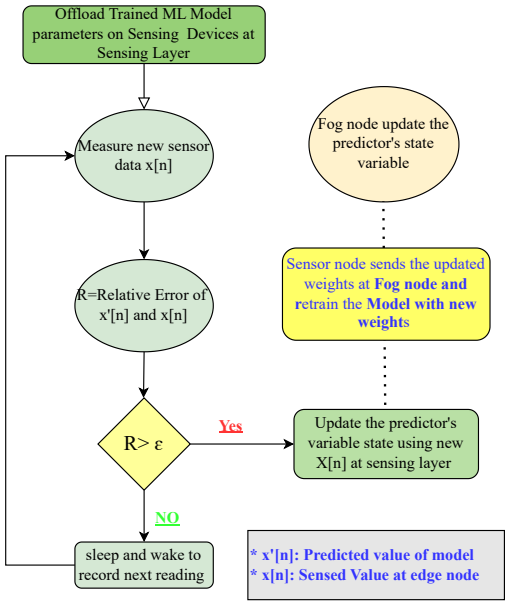


Figure 1: Flowchart representation of proposed model

as well as the Fog layer and there's no need to communicate any data between these layers in usual cases. This nullify the whole data communication and data bandwidth overhead.

A. Comparasion of ML Models

Regression is a predictive modeling technique that analyses the relationship between targeted, and dependent and independent variables of a data set. In this work, we experimented with various regression models like **Linear Regression**, **Ridge Regression**, **Regressive Decision tree**, **Support Vector Machines (SVM)** and **Ensemble Trees**. The selection of the model is carried out based on criteria like Compression size upon deploying on sensors, and model accuracy. From Table

Model	Compressed Size (in KBs)	Accuracy(in %)
Linear Regression	0.545	99.1344
Ridge regression	0.503	99.1344
SVM	778.24	98.8371
Desicion tree	311.296	99.9758
Ensemble tree	770.048	99.4773

Table I: Models Specs Comparison

I, Linear Regression and Ridge Regression show very low compressed disk size after deploying on the sensors, while we get the highest accuracy of 99.9758% in Decision Tree. Later in Section III-D2, we discussed other parameters on which the model's performance was evaluated.

B. Dataset Selection and Preparation

The input data for the model from is collected from the year 2019 - 2020[1]. This data has been used as an External dataset

[1]http://smartfasal.in/ftp-dataset-portal/

to train and test our Model. The data range from 3 March to 16 April, of which the 80% of the whole dataset has been used for training and the rest is used for testing. The dataset has the following attributes: Time, sm1 (Soil Moisture 1), sm2 (Soil Moisture 2), T (Temperature), H (Humidity), and P (Pressure). A correlation matrix is formed to define the attributes that will be provided for model training. The correlation depicts that there is a high correlation of target variables with temperature and humidity. After that, the model starts predicting the targeted attributes. The model is then tested for the rest 20%, for accuracy instances and performance evaluation. In the later sections, we will discuss the performance of the models on live data that has been recorded in the dynamic environment to depict the practical compatibility of the proposed algorithm.

C. Deployment of Model

Another major challenge is to deploy the same model at the fog and edge layer. At the fog level, the model can be deployed with no ramifications since it has enough processing abilities and storage. But, to do the same at the Sensor layer it a challenging task. The devices don't have enough processing power to handle heavy Machine learning models. To alter this problem the model is compressed to flat buffer files using tensor flow and then those lightweight files are deployed to the edge devices. After these two models are deployed successfully both models are tested for synchronous predictability and the same is shown in Figure 2.

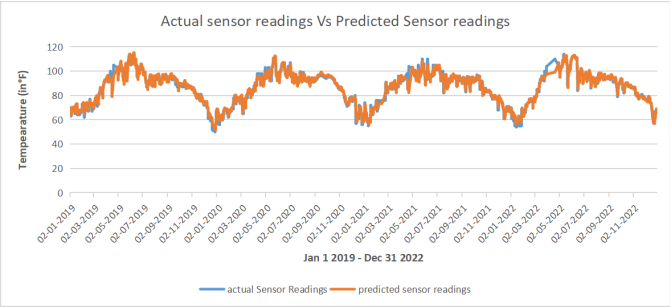


Figure 2: Prediction accuracy

From Figure 2, we can infer that the model is running in synchronisation at both the fog Layer and edge Layer. Also, the actual readings are inclined with the predicted readings. As the model, is deployed, therefore there is no need to physically send the data from edge to fog. If some discrepancies like crossing from the thresholds occur between the actual data and predicted data at the edge layer, then weights and α (learning rate) and other hyper-parameters will be tuned and the values will be modified at both edge and fog levels to retrain the model and retract the architecture.

Now, since the model is effectively predicting the sensor data at both the layers, thus, if any anomaly occurs at the sensing layer such that the **mean absolute error** and **mean relative error** superseded the set threshold ε, then the model needs to retrain itself. As the actual data is received only at the edge layer, the model at the fog layer will not know about the anomalous data. So, the model at the fog level also needs to retrain itself. Therefore, we need to communicate a weight matrix from the edge level to the fog level to retrain the model. To perform this communication, we used Long Range data transmission technique. The Sensors at the edge level are equipped with LoRa Module **Sx 1276** that will transmit the matrix, and at the Fog layer a Receiver LoRa Hat Module is placed on a Raspberry Pi which will collect the data. Once the data matrix is received at the fog layer, the model will retrain itself by adjusting the weights and tuning the hyperparameters and values of α. In this manner, we have also obviated the dependencies on the Internet upto bigger extent in our architecture.

D. Algorithm Methodology (RPDM)

Algorithm 1 `RPDM`

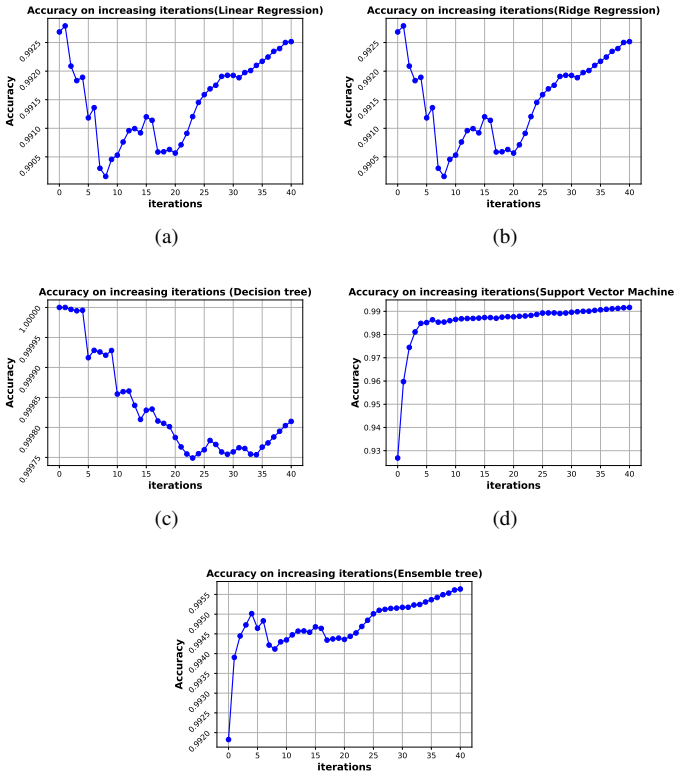
Input: data frame as data $(a_1,b_1,c_1,\dots)$, $(a_2,b_2,c_2,\dots)$ $\dots$ $(a_n,b_n,c_n,\dots)$
Output: Set of clusters

`imports` pandas as pd
`imports` Ridge from sklearn as rr
`Preprocess the data`
$null_{pct} = (data.apply(pd.isnull).sum()/data) * 100$
$valid_{columns} = [null_{pct} < 0.5]$
data=data.ffill()
$corr \leftarrow correlation_{mat}(data)$
`Training and Testing`
start $\leftarrow w_{train}$
step $\leftarrow w_{test}$
threshold $\leftarrow \varepsilon$
for each i in [start, end, step] **do**
    train $\leftarrow$ data[:i,:]
    test $\leftarrow$ data[i:(i+step),:]
    prediction $\leftarrow$ rr.predict(test)
    if $error_{relative}(test, prediction) > threshold$ **then**
        $fog \leftarrow weights \in (a_1{}^*, b_1{}^*, c_1{}^*, \dots)$
    end if
    combined = pd.concat[test,prediction]
    combined.columns= ["actual", "prediction"]
end for
Return *combined*

Table II: Corelation Matrix

	Soil Moisture	Temperature(in °C)	Humidity
Soil Moisture	1	-0.229238	0.401754
Temperatue(in °C)	-0.229238	1	-0.840894
Humidity	0.401754	-0.840894	1
$Target_{attr}$ Temp	-0.230201	0.996164	-0.832561
$Target_{attr}$ Humidity	0.402074	-0.843666	0.994157

The **RPDM** (Regressive Prediction Data Forwarding Model) (Algorithm 1) consists of two phases, **pre-processing of data** and **training and testing of Model**.



(a) (b) (c) (d) (e)

Figure 3: (a) Linear Regression (b) Ridge Regression (c) Decision tree (d) Support Vector Machine (e) Ensemble Tree

1) Pre-processing of data: The data is pre-processed using pandas **api**. Firstly, the data is checked for **NaN** values. The variable $null_{pct}$ as mentioned in Algorithm 1 determines the ratio of null values in each column, and the variable $valid_{columns}$ stores the columns whose $null_{pct}$ value is less than 0.5. Then, columns whose $null_{pct}$ is greater than 0.5 will be discarded. The next step is to get rid of the leftover **NaN** values in $valid_{columns}$ and for that $.ffill()$ API of pandas is used to fill the **NaN** values accordingly. In the Next Step, a Correlation Matrix as shown in Table II is formed as mentioned in variables *corr*, this matrix is imperative for deciding the attributes to predict the target variable's attributes.

2) Testing and Training of Model: At first, we decided on a training and testing window. We decided to make a training window of size w_{train} data points and a testing window of w_{test} data points. Here, a threshold in terms of relative error is set if the relative error crossed this threshold then the weights will be sent across the layers to retrain the model. The model is not trained and tested for the whole data frame at once. It will be trained in the size of N training and testing windows. For example, first, a data frame window of w_{train} will be trained, and then, the model will be tested over for w_{test} data points (testing window size). In this manner, the whole data frame will be trained and tested in an iterative manner. The advantage of this kind of training testing scheme is that the model will periodically train itself. As discussed in section III-A, the same iterative training was performed for all the five methods as shown in Figure 3 and the observed trends and inferences are as follows:

- The Figure 3(a) for Linear Regression and 3(b) for Ridge Regression are identical and also don't show the monotonically increasing and decreasing nature which depicts that the model's performance will be less reliable in the iterations range of 0-20 but when more data is parsed onto the model they both show the improving trend.
- The Figure 3(c) is of a Decision Tree. Initially, the model's performance was better with an accuracy of over 100%, but the model's accuracy trend is monotonically decreasing.
- The Figure 3(d) is of a Support Vector machine (*SVM*), the accuracy characteristic shown by this model is monotonically increasing which shows that upon feeding more data into the model its performance improved.
- Figure 3(e) shows the plot of the Ensemble tree. All the previously trained models were used to train this ensemble tree and the same is evident from the accuracy trend of the model. It encompasses all the Figure 3(a), 3(b), 3(c) and 3(e).

This can be observed in Figure 3, as the number of iterations of model training increases, the model's accuracy varies.

IV. Performance Evaluation

A. On existing Recorded data

In this section the model is tested on an external dataset as mentioned in section III-A. The attributes on which the model is being tested are *Temperature*, *Humidity*, *SoilMoisture*.

The model testing parameters as mentioned in previous section are Accuracy, realtive error and Correlation matrix, extending the parameters as follows.

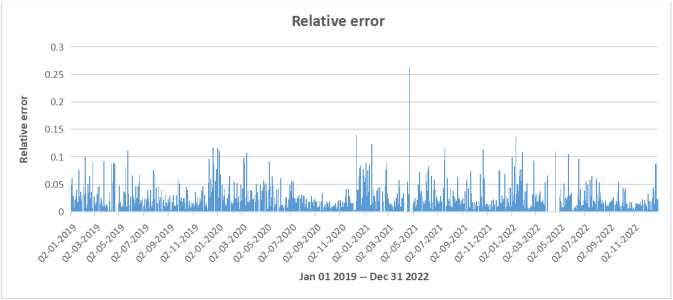


Figure 4: Relative error on existing data

- **The accuracy Measure** The model's Accuracy is being represented in Figure 2. It is evident from the graph that both actual and predicted data points are overlapping with each other, which shows that the model is running in synchronisation with the actual values.
- **Relative Error** The measure of relative error is crucial for our model since, it has an upper limit ϵ and is calculated

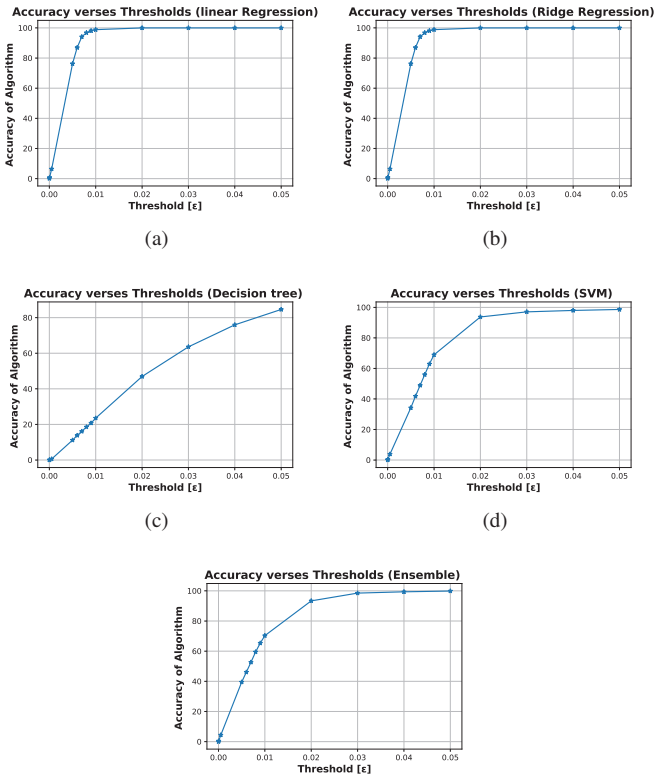


Figure 5: (a) Linear Regression (b) Ridge Regression (c) Decision tree (d) Support Vector Machine (e) Ensemble Tree

using Equation 1, the main idea and motivation of this research are to reduce the data transmission and save bandwidth.

$$Relative_{error}\ \delta = |(V_a - V_e)/V_e| \tag{1}$$

B. On Smart Fasal Dataset

In case of machine learning predictions the maximum percentage relative error should not be above 10% (i.e. below 0.1). For this dataset, from Figure 4, it can be observed that the relative error of the model (in most of the cases) is below 0.1. It shows that the data transmission has been reduced to the negligible range.

Upon varying the values of ε, how the trends of accuracy vary for different models are shown in Figure 5. The plots show that the accuracy of each model increased upon varying the values of threshold. The models show nearly exponential.

C. On Live Data

RPDM algorithm is tested on the live recorded data [9] and was tested on various parameters as discussed in the previous section. While testing the model on a real test bed set up the model needs to be trained with some data on ground truth which is discussed in the next Subsection.

1) Test Bed SetUp and Gaussian Noise: Since the environmental conditions do not vary much considering we are taking for a shorter duration of time, so, a real Test Bed was set up on the field as shown in Figure 6. It consists of Temperature, Humidity and Soil Moisture Sensors being placed on Single arduino boared with LoRa Sx1276 hat enabled on it. The whole is considered to be a single Sensor node. Similarly, two sensor nodes were developed and placed at two different locations on field to collect data. After collecting data for two-month, the same is passed on to the model for training. Similarly, the next 15 days' data was predicted by model on which the testing evaluation of the model was carried out by taking into consideration the next real data from the sensors. Data readings for a day have been predicted by a sensor node and the prediction accuracy for all regressive models was tested. Figure 7, depicts the accuracy precision of all the models.

Figure 6: Test Bed Set up

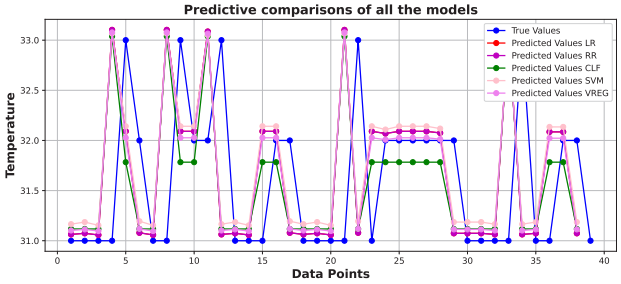


Figure 7: Prediction accuracy for different models

2) Power consumption With and Without Model: As discussed in [10], [11], the authors have worked upon reducing the power consumption between the three layered architecture, on similar note we have also inculcated the power consumption analysis, since it is very crucial for the commercialization of any electronic product. For Calculating the power consumed by the device three power has to be measured, a) Sensing power, b) Transmission Power, and c) Receiving/Processing power. Since the model is continuously running at the sensing layer, so, the Sensing power is constant and the receiver (Fog Node) is also receiving the predicted values continuously, so the Receiving Power will also not play any major role in power flow analysis. The major component will be of transmission power that is from Sensing layer to fog layer. Transmission Power is calculated as given in Equation 2. By taking the same into consideration the model is being tested.

$$Tx_{power} = Path_{loss} + Required_{SNR} - Antennae_{loss} - othergain_{losses} \quad (2)$$

Firstly, the power consumption by the architecture (if no model was there) is tested, and then on increasing values of ε how the power consumption of the model decreases was tested. Since on increasing the values of ε less data need to be transmitted between sensing layer and fog layer as shown in Figure 9. Accordingly, a suitable value for ε was chosen to have the relative error in the permissible range. Figure 8 shows how the power transmission and power consumption was measured by placing a multi-meter and upon increasing the values of ε inside the algorithm how the transmission power changes, which is plotted in Figure 9.

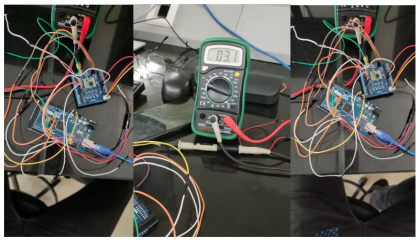


Figure 8: Set Up to Measure Power Consumption of Model

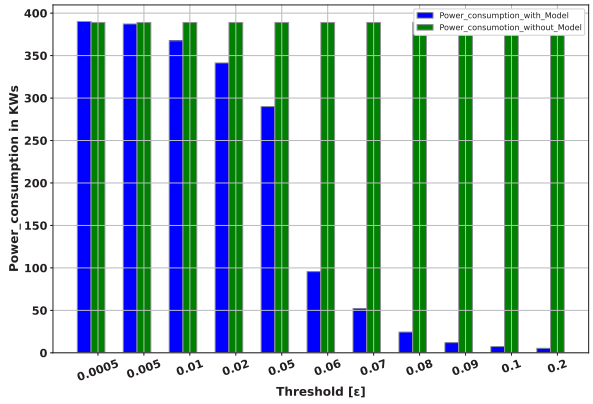


Figure 9: Power Consumption Analysis of Model

D. Discussion

RPDM's architecture was selected on the folowing parameters.

- From Table I it can be evidently noted that the decision tree is performing well in terms of accuracy followed by SVM, Ensemble trees, linear regression, and Ridge Regression.
- From plots description discussed in Section III-D2, Support Vector Machine is performing well above all the models.
- Section IV-C shows that upon increasing the values of ε, SVM shows practically realizable accuracy. This happens as the accuracy is not constant after a certain value as in the case of linear regression and ridge regression. Also, it is not linear as in case of decision tree and in case of Ensemble tree it is an exponential characteristic but it also has a component of Linear and Ridge Regression and Decision Tree since it is not reliable.
- Also we found that, when all the models were tested in and agricultural field then SVM shows very low degree of variability.

V. Conclusion

From the performance of the model while testing both on the external dataset and live data it can be concluded that the **RPDM** model can perform really well in case of data logging in the absence of Internet connectivity, which is a trivial problem in remote locations of the agriculture field. Finally, we tested the proposed framework on a real setup and validated the effectiveness in terms of the amount of data transmitted, and energy consumption using different types of sensors.

Acknowledgment

We acknowledge TIH-IoT at IIT Bombay for sponsoring this work through Technology Development Program via sanctioned letter no. TIH-IoT/21-22/ TDP/Acad/SL/004.

References

[1] Y. Mekonnen, S. Namuduri, L. Burton, A. Sarwat, and S. Bhansali, "Machine learning applications in iot based on agriculture and smart farming: A review," *International Journal of Engineering Applied Sciences and Technology*, vol. 4, no. 12, pp. 24–27, 2020.

[2] M. Vyas, G. Anand, R. N. Yadav, and S. K. Nayak, "Dasa: An efficient data aggregation algorithm for lora enabled fog layer in smart agriculture," in *Advanced Information Networking and Applications: Proceedings of the 37th International Conference on Advanced Information Networking and Applications (AINA-2023), Volume 2*, pp. 40–52, Springer, 2023.

[3] M. Moh and R. Raju, "Machine learning techniques for security of internet of things (iot) and fog computing systems," in *2018 International Conference on High Performance Computing & Simulation (HPCS)*, pp. 709–715, IEEE, 2018.

[4] D. Glaroudis, A. Iossifides, and P. Chatzimisios, "Survey, comparison and research challenges of iot application protocols for smart farming," *Computer Networks*, vol. 168, p. 107037, 2020.

[5] Y. Mekonnen, S. Namuduri, L. Burton, A. Sarwat, and S. Bhansali, "Machine learning techniques in wireless sensor network based precision agriculture," *Journal of the Electrochemical Society*, vol. 167, no. 3, p. 037522, 2019.

[6] T. Ayoub Shaikh, T. Rasool, and F. Rasheed Lone, "Towards leveraging the role of machine learning and artificial intelligence in precision agriculture and smart farming," *Computers and Electronics in Agriculture*, vol. 198, p. 107119, 2022.

[7] W. X. Bu F., "A smart agriculture iot system based on deep reinforcement learning," *The International Journal of eScience*, 2019.

[8] K. Zhang, Y. Zhou, C. Wang, H. Hong, J. Chen, Q. Gao, and M. Ghobaei-Arani, "Towards an automatic deployment model of iot services in fog computing using an adaptive differential evolution algorithm," *Internet of Things*, p. 100918, 2023.

[9] G. Anand, M. Vyas, R. N. Yadav, and S. K. Nayak, "On reducing data transmissions in fog enabled lora based smart agriculture," *IEEE Internet of Things Journal*, 2023.

[10] M. K. Alasmari, S. S. Alwakeel, and Y. A. Alohali, "A multi-classifiers based algorithm for energy efficient tasks offloading in fog computing," *Sensors*, vol. 23, no. 16, 2023.

[11] K. Fathallah, M. A. Abid, and N. Ben Hadj-Alouane, "Enhancing energy saving in smart farming through aggregation and partition aware iot routing protocol," *Sensors*, vol. 20, no. 10, 2020.