Internet of Things 3–4 (2018) 12–33

Contents lists available at ScienceDirect

Internet of Things

journal homepage: www.elsevier.com/locate/iot

Performance analysis and implementation of an adaptive real-time weather forecasting system

T.P. Fowdur[a], Y. Beeharry[a,*], V. Hurbungs[b], V. Bassoo[a], V. Ramnarain-Seetohul[c], E. Chan Moo Lun[a]

[a] *Department of Electrical and Electronic Engineering,* Faculty of Engineering, *University of Mauritius, Mauritius*
[b] *Department of Software and Information Systems,* Faculty on Information, Communication, and Digital Technologies, *University of Mauritius, Mauritius*
[c] Department of Information, Communication and Technology, Faculty of Information, Communication, and Digital Technologies, *University of Mauritius, Mauritius*

ARTICLE INFO

Article history:
Received 15 August 2018
Revised 31 August 2018
Accepted 1 September 2018
Available online 6 September 2018

Keywords:
Real-time weather forecasting
Internet of things
K-nearest neighbors (K-NN)
Multiple linear regression (MLR)
Adaptive K-NN
Adaptive MLR

ABSTRACT

Recently several real-time weather forecasting systems based on Internet of Things (IoT) have been developed to provide short-term real time forecasts also referred to as Nowcasts. The main challenge in these systems is to use appropriate prediction algorithms that can predict different weather parameters with the highest possible accuracy. In this work, an IoT based weather forecasting system has been implemented to provide short-term weather forecasts on a University campus at intervals ranging from 20 min to one hour. Several adaptive forecasting algorithms based on variants of the Multiple Linear Regression technique as well as K-Nearest Neighbors (K-NN) have been experimented. Moreover, three adaptive selection criteria for selecting the most appropriate prediction algorithm for a given Nowcast have been developed and tested. The parameters analysed are: temperature, humidity, atmospheric pressure, rainfall, luminosity, wind-speed, and wind direction. The best adaptive schemes are able to predict these parameters with an overall percentage error of 6.58% as compared to non-adaptive ones which predicted with a worst case percentage error of 13.59%.

1. Introduction

Weather forecasting has always been a vital process to ensure the smooth running of several important activities [1]. However, due to drastic climatic changes, weather prediction using the traditional techniques is becoming ineffective. Many countries are at risk of flash flood nowadays and predicting such weather conditions with conventional forecasting systems is not possible because these systems provide predictions for large regions over hours. Mauritius has recently experienced drastic weather conditions such as flash-floods that caused major collateral damage and life loss. However, nowadays, with the emergence of Internet of Things, real time or near real time weather prediction is possible. There are several weather forecasting systems based on localized sensors connected to cloud computing facilities that can provide short-term forecasts for small regions. These forecasting systems make use of techniques such as neural networks, Fuzzy logic, time series and regression analysis.

* Corresponding author.
E-mail address: y.beeharry@uom.ac.mu (Y. Beeharry).

https://doi.org/10.1016/j.iot.2018.09.002
2542-6605/

List of Notations

T	Temperature in degrees Celsius
H	% Humidity
L	Light intensity in
R	Rainfall in mm
P	Atmospheric pressure in Pa.
WD	Wind direction converted to a degrees scale.
WS	Wind speed in m/s
W	The training window which was fixed to 120 min in this work.
t	The time index.
Px_t	denotes any other related parameter that could influence the value of $P1_t$.
$a_0(W)$, $a_1(W)$ and $a_2(W)$	are coefficients determined for a training window of size W which was kept at 120 min.
p	represents the order of autoregression,
q	is the order of moving average (number of past error terms),
Ø	is the slope coefficient,
Θ	is the moving average coefficient,
e	is the error term
μ	is a constant term
K	is the neighborhood size
y_k	*is* the nearest reading
T_A	The actual or current time.
T_S	The time at which the first value in the training window was recorded.
T_M	The mid-time interval between T_S and T_A.
T_{A+I}	The time at which the values of the parameter is to be predicted. The symbol I represents an interval in minutes after the actual time T_A.
W	The training window size in minutes which essentially represent a set of pre-observed values of the parameter to be predicted over the past W minutes going back from the actual time T_A to T_0.
V_A	The actual or current values of the parameter being measured at time T_A.
V_S	The value of the parameter being measured at time T_s.
V_M	The value of the parameter being measured at time T_M.
$\hat{V}^x_{A+I}$	The value of the parameter predicted with P^x at time T_{A+I}.
$W/2$	It is equivalent to half the window size W and contains values of the parameter recorded from time T_M to T_A i.e. the values V_M to V_A.
$\hat{V}^x_M$	is a value predicted at time T_M with predictor P^x using available values from V_S to V_{M-1}.
$\hat{V}^x_{M+1}$	is a value predicted at time T_{M+1} with predictor P^x using available values from V_S to V_M.
$\hat{V}^x_{M+i}$	is a value predicted at time T_{M+i} with predictor P^x using available values from V_S to V_{M+i-1}. The index i takes values from 1 to $W/2$.
$SE^x_M, SE^x_{M+1}, SE^x_{M+2}, \ldots, SE^x_{M+i}, \ldots, SE^x_A$	represent the squared Euclidean distances between the actual values $V_M, V_{M+1}, V_{M+2}, \ldots V_{M+i} \ldots V_A$ and $\hat{V}^x_M, \hat{V}^x_{M+1}, \hat{V}^x_{M+2}, \ldots, \hat{V}^x_{M+i}, \ldots, \hat{V}^x_A$ respectively.

Numerous IoT based solutions have been implemented to observe weather conditions at specific locations [2–5]. Most systems monitored environmental conditions such as temperature, precipitation, relative humidity, light intensity and CO_2 level. The architectures consisted of a series of sensors connected to a microcontroller platform such as Arduino or Raspberry Pi. The data collected were sent to either a computer or a cloud computing platform for storage and processing. Different methods were used to make the data available to users of the system. In [2], tweets were used to disseminate the gathered information and followers of the twitter account received information regarding daylight, temperature and humidity. In [6–8], flash floods monitoring system were implemented to provide timely information to threatened population and concerned authorities when the water level rises beyond the predefined threshold value. SMS was used to send alert messages [6,8]. Water quality classification using Pollution Index method has been performed using linear SVM and decision tree algorithms in [9] for the preservation of the water ecosystem in maritime and archipelagic countries. A flood prediction system was also implemented using water level and velocity of the water [6]. In [10], the authors introduced CloudCast, a mobile application for localized short-term weather prediction. CloudCast consists of an architecture linking weather radars to cloud resources and a Nowcasting algorithm that accurately predicts short term weather conditions. According to the

authors, CloudCast was able to perform accurate prediction with less than 2 min delay to deliver 15-min Nowcast image to a mobile client.

Several algorithms have been used for weather forecasting. Data collected from sensors are fed into algorithms to make predictions. The algorithms used include regression–based prediction where in [11] data was gathered locally at Pantnagar Station and processed to obtain statistical measures of the hidden information. Furthermore, in [12] forecasting of seasonal and annual rainfall was done by using a nonlinear modeling with Gamma Test (GT) in north of Iran. Rainfall was also predicted using modified linear regression in different regions of Southern India in [13]. Also, Hadoop-based ARIMA algorithm has been used to forecast the data of the coming fortnight by taking data collected over the past decade in [14]. The ARIMA model was also used to forecast wind speed for a period of 7 days in Latvia [15] and to forecast weather conditions for the Abadeh region of Iran [15]. The Seasonal ARIMA model was used for forecasting of monthly rainfall and temperature for monthly intervals for a region of India in [16]. In [17] the ARIMA model was applied for weather prediction in Heipang airport- Jos Plateau, Nigeria where variables used for the study were temperature, rainfall and wind speed. The ARIMA model was also studied in [18] and a comparison was made between ARIMA and the Artificial Neural Network (ANN) to predict rainfall in Mauritius. In [19] a comparative study of the most recent computational intelligence techniques which have been applied for meteorological time series prediction purpose was made. A modified version of the K-NN approach was applied to predict weather data such as solar radiation, maximum and minimum temperature, and rainfall in [20]. In [21] a model for the prediction of weather and solar units based on the K-NN algorithm was used. In [22], K-NN was used with an approach of data mining for the implementation of the prediction system. In [20] the prediction of daily weather data for climate forecasts using the non-parametric nearest-neighbor re-sampling technique was applied. A comparative study of Moving Average on Rainfall Time Series Data for Rainfall Forecasting Based on Evolving Neural Network Classifier was utilized in [23]. Also, the Artificial Neural Network was applied for weather forecasting in [24–26]. In [27], prediction of the Dutch weather was done using recurrent neural networks. In [28], the neural network training model was used for weather forecasting using fireworks algorithm.

In this paper, an IoT based short-term weather prediction architecture is implemented. The main novelty of the paper is the formulation of three variants of the Multiple Linear Regression (MLR) algorithm which are obtained experimentally to determine the combination of parameters yielding the best performance. Moreover, adaptive versions of the MLR and K-NN algorithms were developed by the use of a time-varying training window. Finally, three adaptive selection criteria were integrated in the system to select the most appropriate prediction algorithm to be used for each real-time forecast. The system was used to capture weather parameters such as temperature, humidity, atmospheric pressure, rainfall, luminosity, wind-speed, and wind direction for a period of eight days on the campus of the University of Mauritius. The best adaptive schemes are able to predict these parameters with an overall percentage error of 6.58% as compared to non-adaptive ones which predicted with a worst case percentage error of 13.59%.

The rest of the paper is organized as follows: Section 2 presents a detailed explanation of the system architecture. Section 3 provides an overview of forecasting algorithms. Section 4 presents the results accompanied by comprehensive analysis and finally the work is concluded in Section 5.

2. System architecture

This section describes the Hardware Set-up, configuration of the micro-controllers and the database server.

2.1. Hardware set-up

The block diagram of the proposed weather forecasting system is shown in Fig. 1.

The weather forecasting system consists of one weather-monitoring node. The node is capable of measuring temperature, atmospheric pressure, luminosity, humidity, wind direction, wind speed and rain through the weather meter and the temperature and humidity sensor. The sensors are connected to the weather shield, which is in turn stacked onto the Arduino micro-controller. The XBee Shield and Wi-Fi modules allow the Arduino micro-controllers to be connected to the Raspberry Pi 3 wirelessly.

The Raspberry Pi 3 is a powerful microcontroller which is used to aggregate the data from the weather-monitoring node and acts as a gateway to the local database server. The Raspberry Pi 3 also functions as an edge IoT device which can perform various processing tasks at an early stage so as to offload the processing burden from the application server alone. A straight forward integration of a Java program can be run on the Raspberry Pi 3 to perform data normalization, data error detection, missing data detection and aggregation of data from several nodes transmitting data in parallel. The Raspberry Pi 3 is connected to the database server through an Internet Router. The application server allows the user of the system to query the database or receive alerts. Typically, the user queries the database using a mobile phone.

The values recorded by the sensors of the weather monitoring node are temperature, humidity, atmospheric pressure, luminosity, wind speed, wind direction, and rainfall. Fig. 2 illustrates how the sensor values are recorded and transmitted. The sensor values obtained from the node is stored in JSON format which is a standard way of storing variable names and their corresponding values in key-value pairs, and transmitted through the wireless link to the Raspberry Pi for processing and storage.

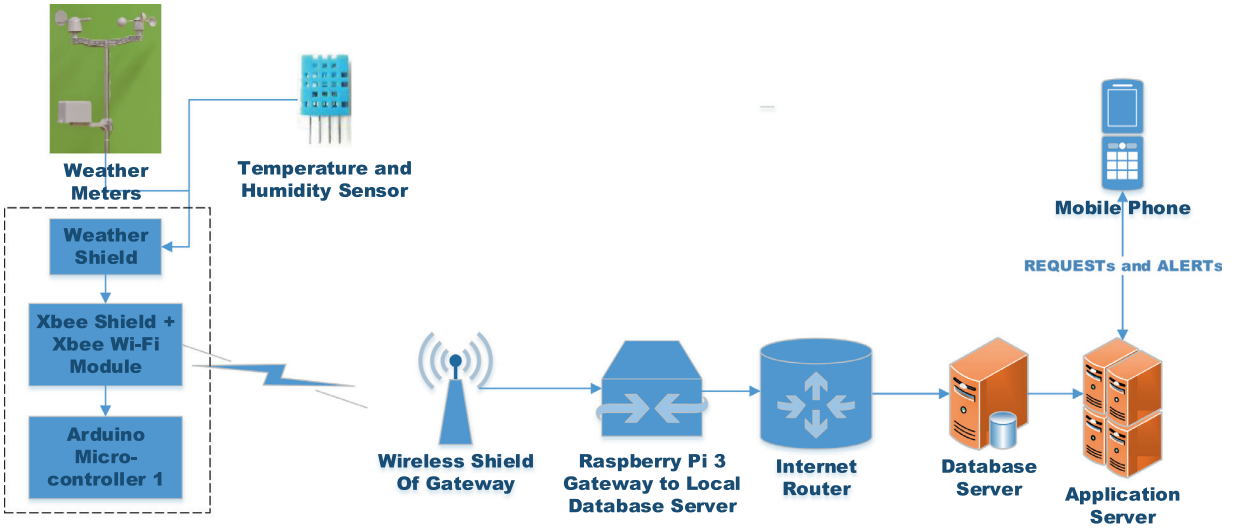

Fig. 1. Block diagram of the proposed weather forecasting system.

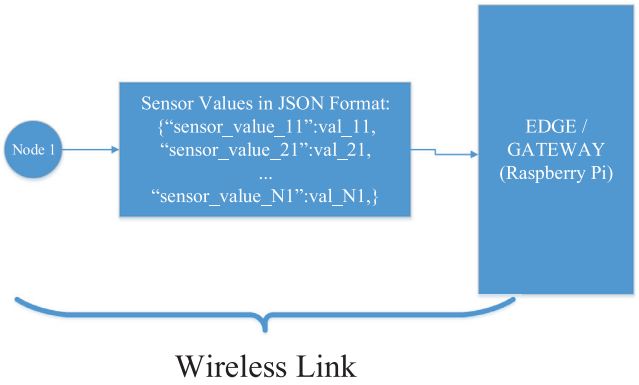

Fig. 2. Sensor values recorded.

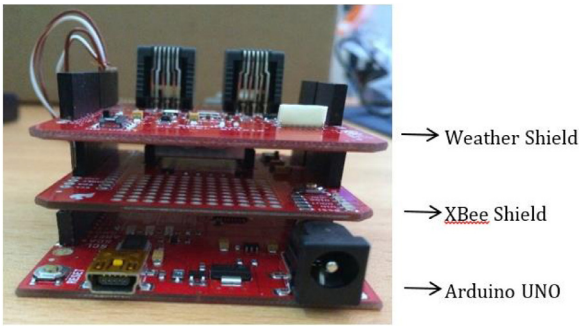

Fig. 3. XBee Shield setup with Arduino Uno and Weather Shield.

Fig. 3 shows how the XBee shield has been used in the project. The XBee shield is mounted on the Arduino Uno and the weather shield is mounted onto the latter. The Xbee Shield allows data from the weather shield and meters to be transmitted to the Raspberry Pi 3 using Wi-Fi.

The XBee and weather shields are mounted directly onto the Arduino. Since both shields use pins 2 and 3 by default; the XBee shield uses them for communication whereas the weather shield uses them to collect the data from the wind and rain sensors. Therefore, a re-wiring to overcome this overlap was required. The unused pins 4 and 5 on the micro-controller were used for the XBee shield to perform the transceiver operations and the default pins were used with the weather shield. The re-wiring performed is shown in Fig. 4.

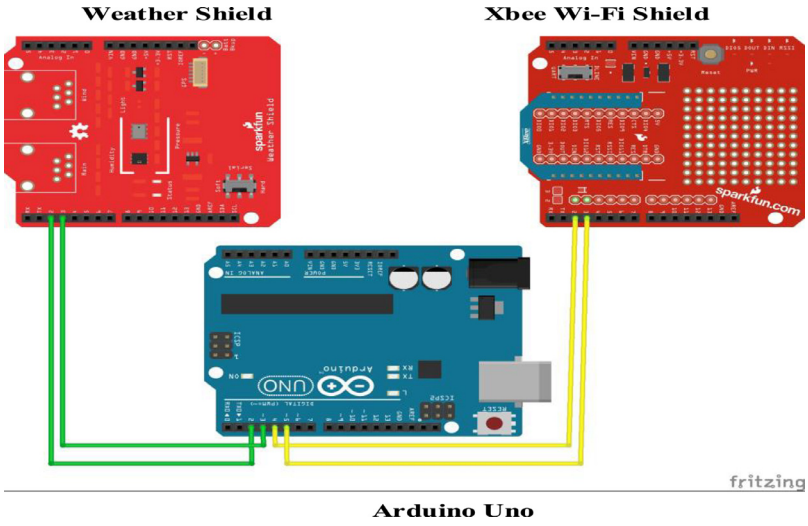

Fig. 4. Re-wiring between XBee Shield, Arduino Uno and Weather Shield.

2.2. Micro-controller configuration and programming

The processes involved in the configurations of the Arduino micro-controller and the Raspberry Pi-3 are explained in the following sub-sections.

2.2.1. Arduino configuration and programming

The Arduino is programmed using the C/C++ programming language whereby configurations have been made to:

- Read data from the different weather sensors,
- Combine all sensor variables and corresponding sensor values in a JSON string,
- Transmit the sensor data in JSON format through the XBee Wi-Fi module on a specific socket connection to be received by the Raspberry Pi-3.

2.2.2. Raspberry Pi-3 configuration and programming

The Raspberry Pi-3 has a Linux-based operating system (Raspbian Jessy) mounted on it and files are amended to configure the device as a wireless router. Java programs are also run for the interception of the data on the socket connection through the wireless interface and inserting it to a specific table in a database server through the Ethernet interface.

2.3. Web server architecture and for data storage and acquisition

The MySQL database which comes along the XAMPP software is used extensively in this work for the local storage of recorded sensor data. These values are then accessed by the Java application for performing predictive analytics and compute forecasted weather data. The latter is then inserted into a table in the same MySQL database. The predicted values can be queried using an application running on a mobile phone. The whole process works is as shown in Fig. 5.

2.3.1. Database architecture

Two tables are implemented in a single database for the proposed system. One table is used for storing the aggregated sensor values and the second one is used to save predicted values which are accessed by the mobile application.

The fields for “id” and “time” are automatically generated and added when new sets of sensor values are written to the table in the database. The “time” field refers to the timestamp at which the data was collected and inserted in the database. The fields “windGustmph”, “windGustdir”, “humidity”, “temperature”, “rainIn”, “pressure”, and “lightLevel” refer to the sensor values of wind speed, wind direction, humidity, temperature, rain level, atmospheric pressure, and luminosity respectively. A sample of the values recorded in the table is as depicted in Fig. 6.

The MySQL database used for this work is a low complexity database as it involves simpler configurations or setup. However, an upgrade to NoSQL databases (Cassandra or MongoDB) would be more appropriate when deploying the system on a larger scale. The advantage of NoSQL in this case would be presented due to large volumes of data being dealt with.

Data from sensors are sent in real-time to the database. The system will obtain only the parameters for the forecasting algorithm and data for a window corresponding to the last 120 min to perform the forecasting. As such, the burden on the system is constant and determined by the window size set and not all the data found in the database. Hence, it will not be an issue to deploy in larger areas. Moreover, to move to a wider area, we will need additional sensor nodes to offer a wider coverage.

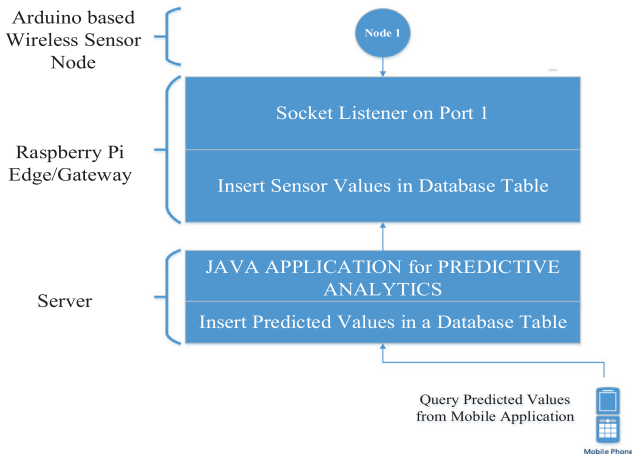

Fig. 5. Architecture for data acquisition and query from mobile device.

id	time	windGustmph	windGustdir	humidity	temperature	rainIn	pressure	lightLevel
22368	2017-11-28 10:34:31	6.4	157.5	40.5	30	0	98173.8	3.195
22367	2017-11-28 10:34:25	6.4	157.5	40.5	29.5	0	98150.2	3.195
22366	2017-11-28 10:34:20	6.4	157.5	40.5	29.5	0	98112	3.195
22365	2017-11-28 10:34:15	6.4	157.5	40	29.5	0	98076.6	3.185
22364	2017-11-28 10:34:09	6.4	157.5	40	29.5	0	98080	3.185
22363	2017-11-28 10:34:04	6.4	157.5	39.5	29.5	0	98082.9	3.185
22362	2017-11-28 10:33:59	6.4	157.5	40	29.5	0	98080.9	3.185
22361	2017-11-28 10:33:53	6.4	157.5	40	30	0	98075.9	3.18

Fig. 6. Sample of Recorded Sensor values.

3. Forecasting algorithms and software design

This section gives an overview of the forecasting algorithms that were investigated in this work.

3.1. Forecasting algorithms

In this section the following general symbols have been used to describe the different variables in the equations:

T	temperature in degrees Celcius
H	% Humidity
L	Light intensity in
R	Rainfall in mm
P	Atmospheric pressure in Pa.
WD	Wind direction converted to a degrees scale.
WS	Wind speed in m/s
W	The training window which was fixed to 120 mins in this work.
t	The time index.

3.1.1. Multiple linear regression

In this work, with multiple linear regression, equations involving a general parameter $P1$ to be predicted at time $t+1$ with respect to the previously recorded value of $P1$ at time t and another related parameter Px were formulated as per the following generic combination:

$$P1_{t+1} = a_0(W) + a_1(W)P1_t + a_2(W)Px_t \tag{1}$$

where,

Px_t denotes any other related parameter that could influence the value of $P1_t$.

$a_0(W)$, $a_1(W)$ and $a_2(W)$ are coefficients determined for a training window of size W which was kept at 120 min.

These coefficients were recomputed after every 120 min to adjust the model to the newly recorded values obtained from the weather sensors.

Three different combinations of multiple linear regression equations denoted as MLR, MLR1 and MLR2 were derived experimentally for the seven weather parameters. These combinations are described as follows:

MLR. With this model, the following combination of equations were used:

$$T_{t+1} = a_0(W) + a_1(W)T_t + a_2(W)H_t \tag{2}$$

In Eq. (2), the value of the temperature for the next time interval, i.e. T_{t+1} is predicted from the previous temperature value T_t and previous humidity value H_t. The constants $a_0(W)$, $a_1(W)$, and $a_2(W)$ are obtained from the regression model for a training window of size W. The details of the equations (3)–22) follow the same convention as for Eq. (2) with their respective parameters.

$$H_{t+1} = a_0(W) + a_1(W)H_t + a_2(W)T_t \tag{3}$$

$$L_{t+1} = a_0(W) + a_1(W)L_t + a_2(W)T_t \tag{4}$$

$$P_{t+1} = a_0(W) + a_1(W)P_t + a_2(W)T_t \tag{5}$$

$$R_{t+1} = a_0(W) + a_1(W)R_t + a_2(W)T_t \tag{6}$$

$$WS_{t+1} = a_0(W) + a_1(W)WS_t + a_2(W)T_t \tag{7}$$

$$WD_{t+1} = a_0(W) + a_1(W)WD_t + a_2(W)T_t \tag{8}$$

MLR 1.

$$T_{t+1} = a_0(W) + a_1(W)T_t + a_2(W)L_t \tag{9}$$

$$H_{t+1} = a_0(W) + a_1(W)H_t + a_2(W)L_t \tag{10}$$

$$L_{t+1} = a_0(W) + a_1(W)L_t + a_2(W)H_t \tag{11}$$

$$P_{t+1} = a_0(W) + a_1(W)P_t + a_2(W)H_t \tag{12}$$

$$R_{t+1} = a_0(W) + a_1(W)R_t + a_2(W)H_t \tag{13}$$

$$WS_{t+1} = a_0(W) + a_1(W)WS_t + a_2(W)H_t \tag{14}$$

$$WD_{t+1} = a_0(W) + a_1(W)WD_t + a_2(W)H_t \tag{15}$$

MLR 2.

$$T_{t+1} = a_0(W) + a_1(W)T_t + a_2(W)P_t \tag{16}$$

$$H_{t+1} = a_0(W) + a_1(W)H_t + a_2(W)P_t \tag{17}$$

$$L_{t+1} = a_0(W) + a_1(W)L_t + a_2(W)P_t \tag{18}$$

$$P_{t+1} = a_0(W) + a_1(W)P_t + a_2(W)L_t \tag{19}$$

$$R_{t+1} = a_0(W) + a_1(W)R_t + a_2(W)L_t \tag{20}$$

$$WS_{t+1} = a_0(W) + a_1(W)WS_t + a_2(W)L_t \tag{21}$$

$$WD_{t+1} = a_0(W) + a_1(W)WD_t + a_2(W)L_t \tag{22}$$

3.1.2. Autoregressive integrated moving average (ARIMA) [29]

ARIMA is a stochastic time series model first popularized by Box and Jenkins [30]. An ARIMA model is a generalization of an Autoregressive Moving Average (ARMA) model that is used to predict a value in time series data as a linear combination of its past values and past errors. The model is used on time series data which can be made stationary by differencing [31]. The general expression of ARIMA forecasting is given below:

$$Y_{i+1} = \mu + \phi_1 Y_{i-1} + \cdots + \phi_p Y_{i-p} + \theta_1 e_{i-1} + \cdots + \theta_q e_{i-q} \tag{23}$$

where,

- p represents the order of auto-regression
- q is the order of moving average (number of past error terms)
- ϕ is the slope coefficient
- θ is the moving average coefficient
- e is the error term
- μ is a constant term

3.1.3. K-nearest neighbors (K-NN) [29]

The K-NN algorithm depends only on the given data set and a user-defined constant parameter namely K. The following procedures explain the K-NN algorithm:

1. With a distance function, the K readings nearest to the prediction point (next interval) is retrieved.
2. The forecasted output is computed as the mean of the K nearest readings given in the following equation:

$$Y_{i+1} = \frac{1}{K} \sum_{i=1}^{K} y_i \tag{24}$$

where,

- K is the neighborhood size.
- y_i *is* the nearest reading.

3.1.4. Adaptive multiple linear regression and K-NN

In order to make the MLR, MLR1 and MLR2 algorithms adaptive, the training window W, used to compute their coefficients a_0, a_1 and a_2 should be updated each time a new parameter is read from the sensors. In other words every t seconds another set of coefficients are computed for the MLR model. The main modification is that the coefficients are now a function of W_t i.e. a training window of size W which is changed every t seconds instead of a fixed training window W. The modified equations are as follows:

Adaptive MLR. With this model, the following combination of equations were used:

$$T_{t+1} = a_0(W_t) + a_1(W_t)T_t + a_2(W_t)H_t \tag{25}$$

$$H_{t+1} = a_0(W_t) + a_1(W_t)H_t + a_2(W_t)T_t \tag{26}$$

$$L_{t+1} = a_0(W_t) + a_1(W_t)L_t + a_2(W_t)T_t \tag{27}$$

$$P_{t+1} = a_0(W_t) + a_1(W_t)P_t + a_2(W_t)T_t \tag{28}$$

$$R_{t+1} = a_0(W_t) + a_1(W_t)R_t + a_2(W_t)T_t \tag{29}$$

$$WS_{t+1} = a_0(W_t) + a_1(W_t)WS_t + a_2(W_t)T_t \tag{30}$$

$$WD_{t+1} = a_0(W_t) + a_1(W_t)WD_t + a_2(W_t)T_t \tag{31}$$

Adaptive MLR 1

$$T_{t+1} = a_0(W_t) + a_1(W_t)T_t + a_2(W_t)L_t \tag{32}$$

$$H_{t+1} = a_0(W_t) + a_1(W_t)H_t + a_2(W_t)L_t \tag{33}$$

$$L_{t+1} = a_0(W_t) + a_1(W_t)L_t + a_2(W_t)H_t \tag{34}$$

$$P_{t+1} = a_0(W_t) + a_1(W_t)P_t + a_2(W_t)H_t \tag{35}$$

$$R_{t+1} = a_0(W_t) + a_1(W_t)R_t + a_2(W_t)H_t \tag{36}$$

$$WS_{t+1} = a_0(W_t) + a_1(W_t)WS_t + a_2(W_t)H_t \tag{38}$$

$$WD_{t+1} = a_0(W_t) + a_1(W_t)WD_t + a_2(W_t)H_t \tag{39}$$

Adaptive MLR 2

$$T_{t+1} = a_0(W_t) + a_1(W_t)T_t + a_2(W_t)P_t \tag{40}$$

$$H_{t+1} = a_0(W_t) + a_1(W_t)H_t + a_2(W_t)P_t \tag{41}$$

$$L_{t+1} = a_0(W_t) + a_1(W_t)L_t + a_2(W_t)P_t \tag{42}$$

$$P_{t+1} = a_0(W_t) + a_1(W_t)P_t + a_2(W_t)L_t \tag{43}$$

$$R_{t+1} = a_0(W_t) + a_1(W_t)R_t + a_2(W_t)L_t \tag{44}$$

$$WS_{t+1} = a_0(W_t) + a_1(W_t)WS_t + a_2(W_t)L_t \tag{45}$$

$$WD_{t+1} = a_0(W_t) + a_1(W_t)WD_t + a_2(W_t)L_t \tag{46}$$

Adaptive K-NN. For adaptive K-NN also the training window is changed every t seconds and the modified equation is as follows:

$$Y_{t+1} = \frac{1}{K}\sum_{t=1}^{K} Y_t \ \ for\ K \in W_t \tag{47}$$

3.1.5. Proposed adaptive selection algorithms

Three adaptive selection algorithms, denoted as A1, A2 and A3 respectively have been developed. The rationale behind these adaptive algorithms is to select the best prediction algorithm for predicting a given parameter based on the predictor that achieved the best performance for the previous prediction. The following selected prediction algorithms were included in the set of prediction algorithms to be used for the adaptive selection algorithms:

1. ARIMA.
2. Adaptive K-NN.
3. Adaptive Multi Linear regression.
4. Adaptive Multi Linear regression 1.
5. Adaptive Multi Linear regression 2.

Let P^x denote anyone of the above predictors where $x = 1, 2, 3, 4, 5$ in this case and P^1 represents Arima, P^2 K-NN and so on.

Consider Fig. 7 and notations that will be used in formulating the prediction algorithms:

The parameters in the above figure are defined as follows:

T_A The actual or current time.
T_S The time at which the first value in the training window was recorded.
T_M The mid-time interval between T_S and T_A.

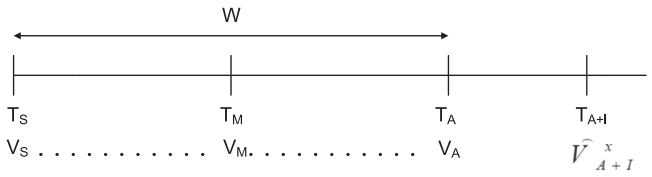

Fig. 7. Timing diagram for Algorithm A1.

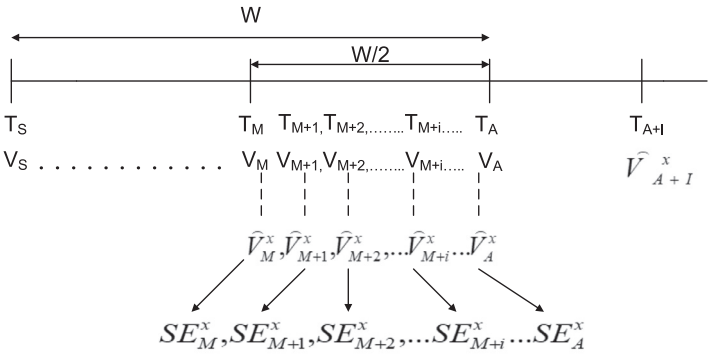

Fig. 8. Timing diagram for Algorithm A2.

T_{A+I} The time at which the values of the parameter is to be predicted. The symbol I represents an interval in minutes after the actual time T_A.
W The training window size in minutes which essentially represent a set of pre-observed values of the parameter to be predicted over the past W minutes going back from the actual time T_A to T_S.
V_A The actual or current values of the parameter being measured at time T_A.
V_S The value of the parameter being measured at time Ts.
V_M The value of the parameter being measured at time T_M.
$\hat{V}^{x}_{A+I}$ The value of the parameter predicted with P^x at time T_{A+I}.

Algorithm A1 is formulated as follows:

1. Using the training set values recorded from time T_S to T_{A-1} i.e. the values V_S to V_{A-1}, predict the value at time T_A with each predictor P^x as follows:

$$\hat{V}^{x}_{A} = P^{x}(V_S, V_{A-1}) \tag{48}$$

where, $\hat{V}^{x}_{A}$ is the value of the parameter predicted at time T_A with a given predictor P^x that uses values from V_S to V_{A-1} to perform the prediction.
More explicitly for the five predictors used in this specific case, there will be five predictions obtained for the values at time T_A as follows:

$$\hat{V}^{1}_{A} = P^{1}(V_S, V_{A-1});\ \hat{V}^{2}_{A} = P^{2}(V_S, V_{A-1});\ \hat{V}^{3}_{A} = P^{3}(V_S, V_{A-1});\ \hat{V}^{4}_{A} = P^{4}(V_S, V_{A-1});\ \hat{V}^{5}_{A} = P^{5}(V_S, V_{A-1}) \tag{50}$$

2. Compute the squared Euclidean distance, between the actual value of the parameter V_A observed time T_A and the predicted value $\hat{V}^{x}_{A}$ obtained with each predictor P^x as follows:

$$SE^{x}_{A} = \left(V_A - \hat{V}^{x}_{A}\right)^2 \tag{51}$$

In this case there will be five values of the MSE obtained as follows:

$$SE^{1}_{A} = \left(V_A - \hat{V}^{1}_{A}\right)^2;\ SE^{2}_{A} = \left(V_A - \hat{V}^{2}_{A}\right)^2;\ SE^{3}_{A} = \left(V_A - \hat{V}^{3}_{A}\right)^2;\ SE^{4}_{A} = \left(V_A - \hat{V}^{4}_{A}\right)^2;\ SE^{5}_{A} = \left(V_A - \hat{V}^{5}_{A}\right)^2 \tag{52}$$

3. Predict the value at time T_{A+I} with the predictor, P^x, giving the lowest MSE for the predicted value at time T_A:

$$\hat{V}^{x}_{A+1} = P^{x}(V_S, V_A)$$
$$x = Index\left(min[SE^{x}_{A}]\right) = Index\left(min\left[SE^{1}_{A}, SE^{2}_{A}, SE^{3}_{A}, SE^{4}_{A}, SE^{5}_{A}\right]\right) \tag{53}$$

Algorithm A2 is formulated as follows:
Consider Fig. 8 which is the timing diagram for algorithm A2.
The following additional terms are defined for the above diagram:

$W/2$: It is equivalent to half the window size W and contains values of the parameter recorded from time T_M to T_A i.e. the values V_M to V_A.

$\overset{\frown}{V}^x_M$ is a value predicted at time T_M with predictor P^x using available values from V_S to V_{M-1}.

$\hat{V}^x_{M+1}$ is a value predicted at time T_{M+1} with predictor P^x using available values from V_S to V_M.

$\hat{V}^x_{M+i}$ is a value predicted at time T_{M+i} with predictor P^x using available values from V_S to V_{M+i-1}. The index i takes values from 1 to $W/2$.

$SE^x_M, SE^x_{M+1}, SE^x_{M+2}, \ldots SE^x_{M+i} \ldots SE^x_A$ represent the squared Euclidean distances between the actual values $V_M, V_{M+1}, V_{M+2}, \ldots . V_{M+i} \ldots . V_A$ and $\overset{\frown}{V}^x_M, \overset{\frown}{V}^x_{M+1}, \overset{\frown}{V}^x_{M+2}, \ldots \overset{\frown}{V}^x_{M+i} \ldots \overset{\frown}{V}^x_A$ respectively.

The algorithm proceeds as follows:

1. Predict the values from time T_M to T_A i.e. $\overset{\frown}{V}^x_M, \overset{\frown}{V}^x_{M+1}, \overset{\frown}{V}^x_{M+2}, \ldots \overset{\frown}{V}^x_{M+i} \ldots \overset{\frown}{V}^x_A$ at a one minute intervals with each predictor P^x as follows:

$$\hat{V}^x_M = P^x(V_S, V_{M-1}) \tag{54}$$

More explicitly:

$$\hat{V}^1_M = P^1(V_S, V_{M-1});\ \hat{V}^2_M = P^2(V_S, V_{M-1});\ \hat{V}^3_M = P^3(V_S, V_{M-1});\ \hat{V}^4_A = P^4(V_S, V_{M-1});\ \hat{V}^5_A = P^5(V_S, V_{M-1}) \tag{55}$$

In general:

$\hat{V}^x_{M+i} = P^x(V_S, V_{M+i-1})$ and:

$$\begin{aligned} &\hat{V}^1_{M+i} = P^1(V_S, V_{M+i-1});\ \hat{V}^2_{M+i} = P^2(V_S, V_{M+i-1});\ \hat{V}^3_{M+i} = P^3(V_S, V_{M+i-1}); \\ &\hat{V}^4_{M+i} = P^4(V_S, V_{M+i-1});\ \hat{V}^5_{M+i} = P^5(V_S, V_{M+i-1}) \end{aligned} \tag{56}$$

2. Compute the squared Euclidean distances $SE^x_M, SE^x_{M+1}, SE^x_{M+2}, \ldots, SE^x_{M+i}, \ldots, SE^x_A$ between the actual values $V_M, V_{M+1}, V_{M+2}, \ldots . V_{M+i} \ldots . V_A$ and $\hat{V}^x_M, \hat{V}^x_{M+1}, \hat{V}^x_{M+2}, \ldots, \hat{V}^x_{M+i}, \ldots, \hat{V}^x_A$ respectively for each algorithm as follows:

$$SE^x_M = \left(V_M - \hat{V}^x_M\right)^2 \tag{57}$$

More explicitly:

$$SE^1_M = \left(V_M - \hat{V}^1_M\right)^2;\ SE^2_M = \left(V_M - \hat{V}^2_M\right)^2;\ SE^3_M = \left(V_M - \hat{V}^3_M\right)^2;\ SE^4_M = \left(V_M - \hat{V}^4_M\right)^2;\ SE^5_M = \left(V_M - \hat{V}^5_M\right)^2 \tag{58}$$

In general:

$SE^x_{M+i} = (V_{M+i} - \hat{V}^x_{M+i})^2$ and more explicitly:

$$\begin{aligned} &SE^1_{M+i} = \left(V_{M+i} - \hat{V}^1_{M+i}\right)^2;\ SE^2_{M+i} = \left(V_{M+i} - \hat{V}^2_{M+i}\right)^2;\ SE^3_{M+i} = \left(V_{M+i} - \hat{V}^3_{M+i}\right)^2; \\ &SE^4_{M+i} = \left(V_{M+i} - \hat{V}^4_{M+i}\right)^2;\ SE^5_{M+i} = \left(V_{M+i} - \hat{V}^5_{M+i}\right)^2 \end{aligned} \tag{59}$$

3. Compute the mean squared error for each predictor for all predictions made from T_M to T_A of the values $\hat{V}^x_M, \hat{V}^x_{M+1}, \hat{V}^x_{M+2}, \ldots, \hat{V}^x_{M+i}, \ldots, \hat{V}^x_A$. For a given predictor P^x taking i from 1 to $W/2$, we obtain the MSE as follows:

$$\overline{MSE}^x = \frac{2}{W}\sum_{i=1}^{W/2} SE^x_{M+1-i} \tag{60}$$

More explicitly:

$$\begin{aligned} &(\overline{MSE})^1 = \frac{2}{W}\sum_{(i=1)}^{(W/2)} SE^1_{(M+1-i)};\ (\overline{MSE})^2 = \frac{2}{W}\sum_{(i=1)}^{(W/2)} SE^2_{(M+1-i)};\ (\overline{MSE})^3 = \frac{2}{W}\sum_{(i=1)}^{(W/2)} SE^3_{(M+1-i)}; \\ &(\overline{MSE})^4 = \frac{2}{W}\sum_{(i=1)}^{(W/2)} SE^4_{(M+1-i)};\ (\overline{MSE})^5 = \frac{2}{W}\sum_{(i=1)}^{(W/2)} SE^5_{(M+1-i)} \end{aligned} \tag{62}$$

4. Predict the value at time T_{A+I} with the predictor, P^x, giving the lowest MSE for the predicted values between T_M and T_A:

$$\begin{aligned} &\hat{V}^x_{A+1} = P^x(V_S, V_A) \\ &x = Index\left(min\left[\overline{MSE}^x\right]\right) = Index\left(min\left[\overline{MSE}^1, \overline{MSE}^2, \overline{MSE}^3, \overline{MSE}^4, \overline{MSE}^x\right]\right) \end{aligned} \tag{63}$$

Algorithm A3 is formulated as follows:

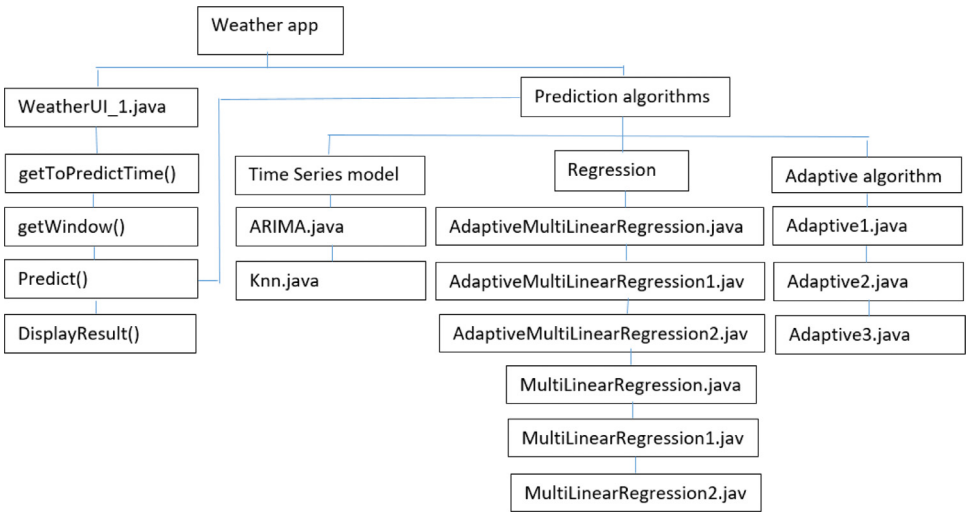

Fig. 9. Program method structure.

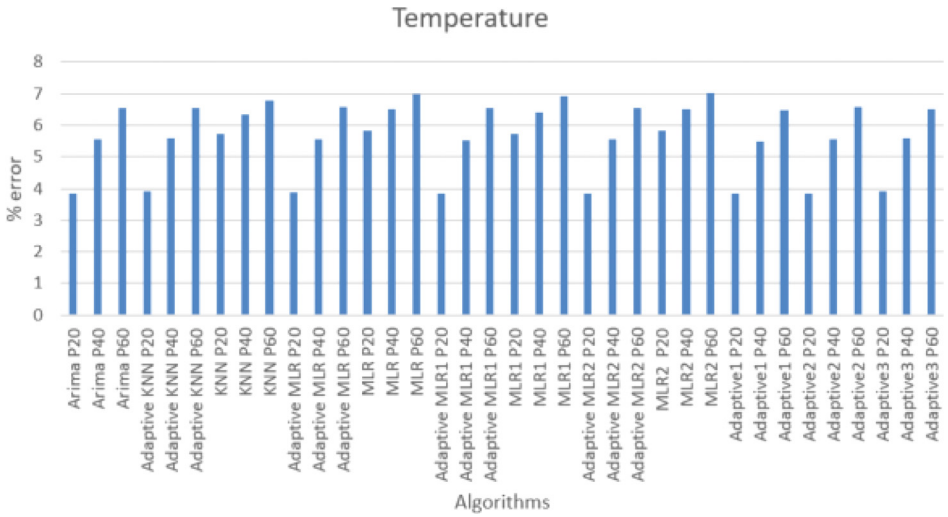

Fig. 10. Percentage error obtained for temperature with the different algorithms.

1. Perform steps 1 and 2 as in Algorithm A2.
2. For each predictor for all predictions made from T_M to T_A of the values $\hat{V}^x_M, \hat{V}^x_{M+1}, \hat{V}^x_{M+2}, \ldots, \hat{V}^x_{M+i}, \ldots, \hat{V}^x_A$, determine the number of times N^x for which the squared Euclidean distance $SE^x_{M+i} = (V_{M+i} - \hat{V}^x_{M+i})^2$ of predictor P^x is the lowest. In this specific case five counts i.e. $N1$, $N2$, $N3$, $N4$, and $N5$, will be obtained for predictors 1–5 respectively.
3. Predict the value at time T_{A+1} with the predictor, P^x, having the highest count Nx for the predicted values between time T_M and T_A:

$$\begin{aligned} \hat{V}^x_{A+1} &= P^x(V_S, V_A) \\ x &= Index(max[N1, N2, N3, N4, N5]) \end{aligned} \tag{64}$$

3.2. Software design

Fig. 9 demonstrates a binary tree organisation of the Java program's methods developed for the application of weather forecasting.

The application consists of twelve Java classes which are made up of one Main Frame class and eleven predictive analysis classes. When the program begins, the Main Frame (MainFrame.Java) runs the graphical components to display the Java form given in Fig. 12. After loading the graphical elements, the WeatherMonitor method extracts the weather information in real-time from the server and displays it on the application. The weather information is then stored in an array structure which is updated continuously every minute. The DisplayResult() function displays the information stored in the array on the user

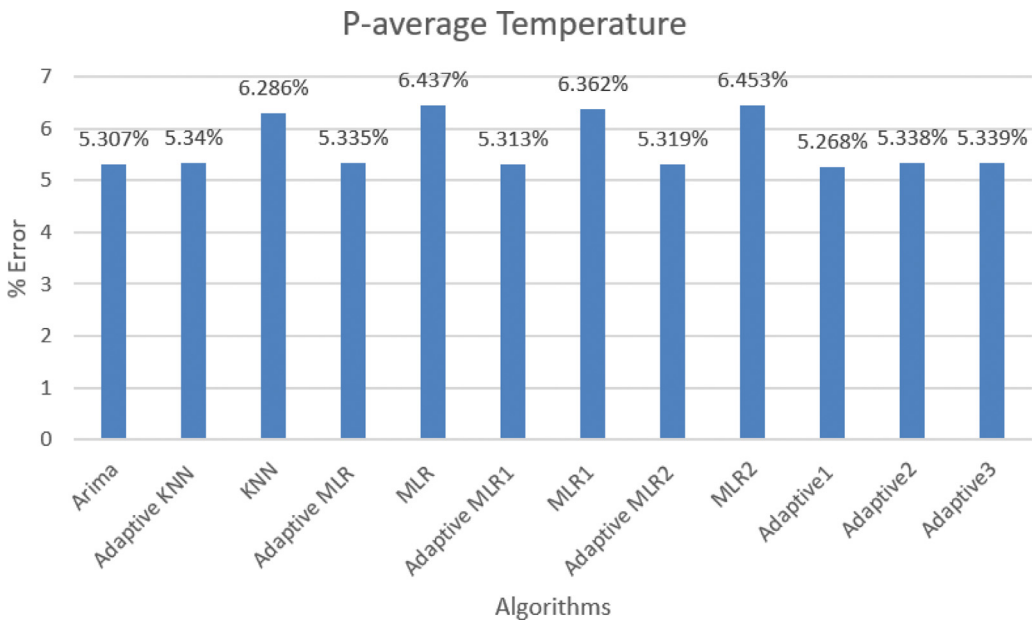

Fig. 11. Average percentage error obtained for temperature over all three intervals with the different algorithms.

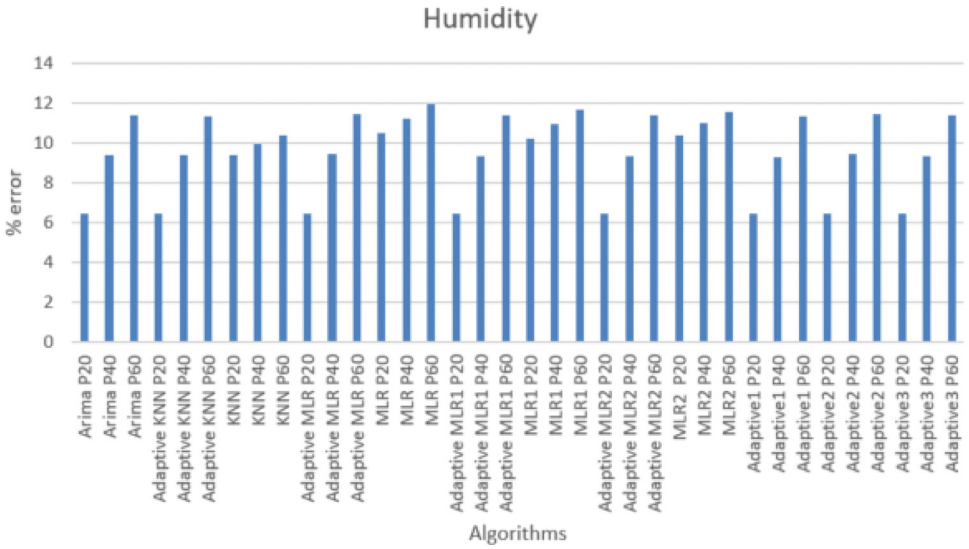

Fig. 12. Percentage error obtained for humidity with the different algorithms.

interface. To perform prediction, the GetWeatherData() method is run to get the weather's recent parameters. The Predict() method uses the prediction algorithms to predict weather. The DisplayPredictResult() method shows the predicted values on the user interface. A user interface was developed to display the predicted values.

4. Results and discussions

The following schemes were tested:

1. K-NN.
2. Adaptive K-NN.
3. Multiple Linear Regression.
4. Multiple Linear Regression 1.
5. Multiple Linear Regression 2.
6. Adaptive Multiple Linear Regression.

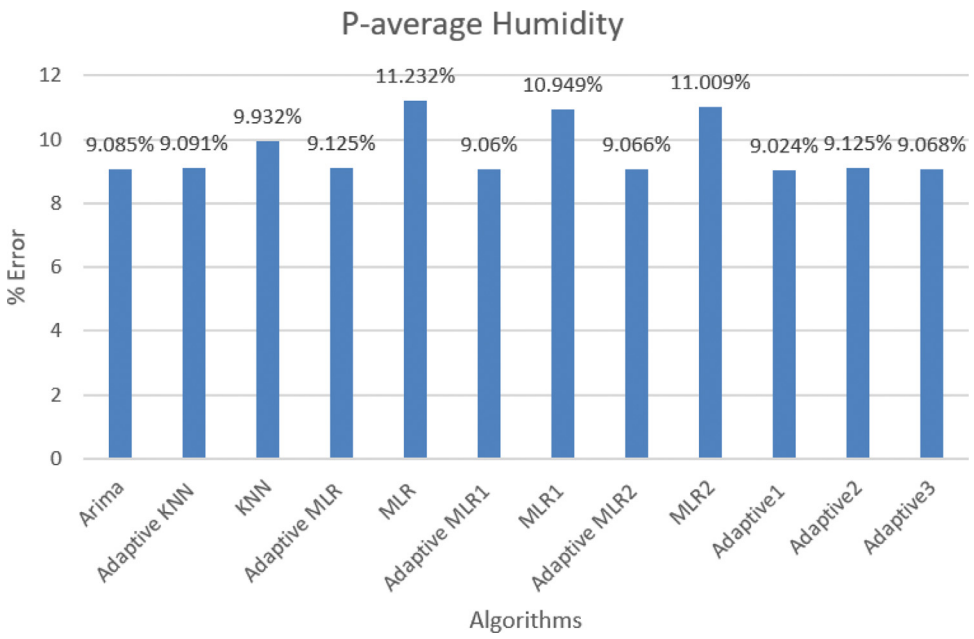

Fig. 13. Average percentage error obtained for humidity over all three intervals with the different algorithms.

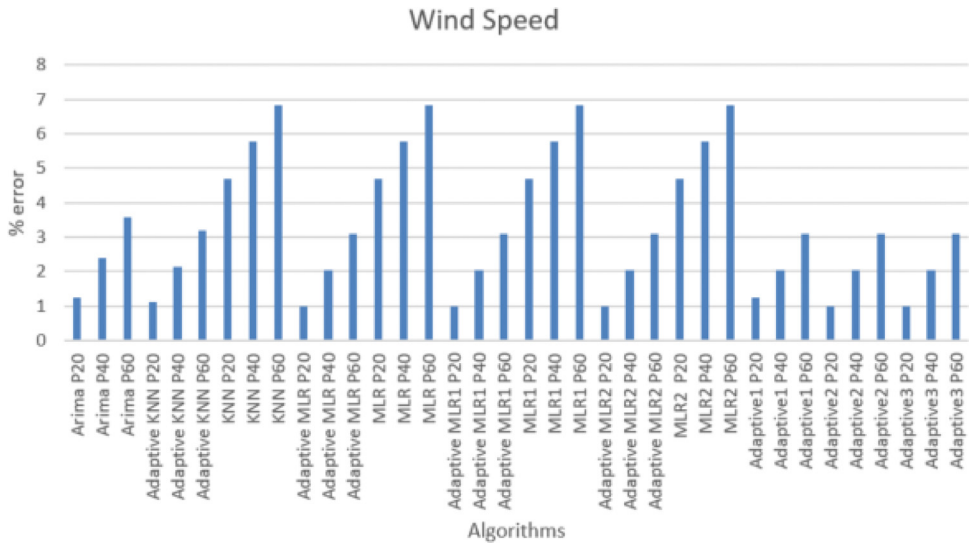

Fig. 14. Percentage error obtained for wind speed with the different algorithms.

7. Adaptive Multiple Linear Regression 1.
8. Adaptive Multiple Linear Regression 2.
9. Adaptive 1 (Adaptive Selection Algorithm 1).
10. Adaptive 2 (Adaptive Selection Algorithm 2).
11. Adaptive 3 (Adaptive Selection Algorithm 3).

The tests were conducted on the UoM campus from 23rd February 2018 to 5th March 2018 with data collected from 0900 to 1600 on each day and predictions made between 1200 and 1600. A window of 2 h was used as a training dataset to predict at intervals of 20 min (P20), 40 min (P40) and 60 min (P60). Readings for the following parameters were obtained at rate of 12 values per minute:

1. Temperature.
2. Humidity.
3. Wind speed.

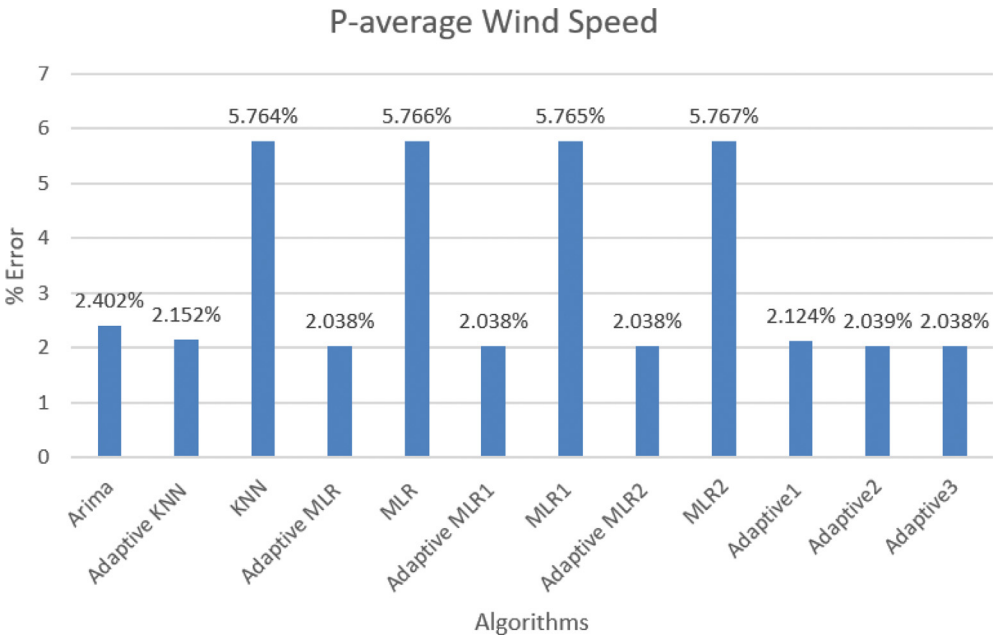

Fig. 15. Average percentage error obtained for wind speed over all three intervals with the different algorithms.

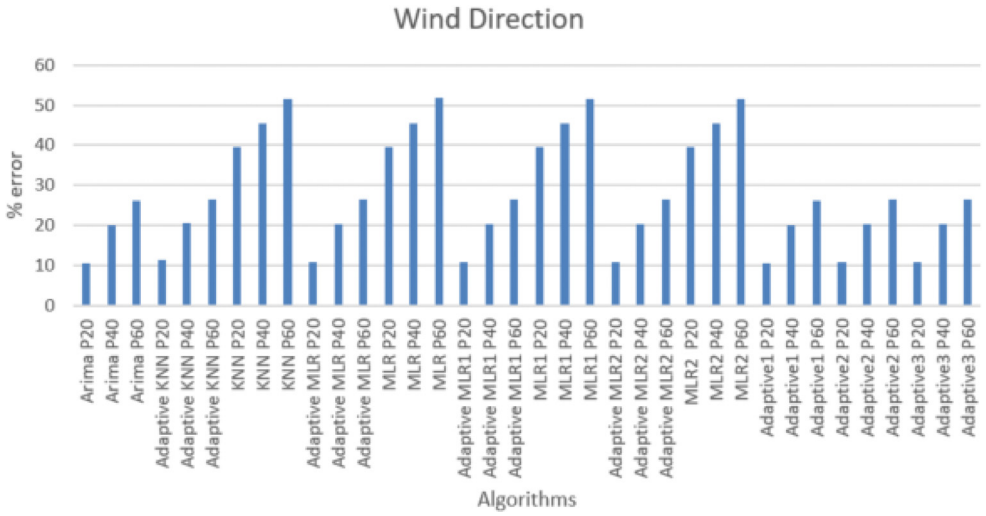

Fig. 16. Percentage error obtained for wind direction with the different algorithms.

4. Wind Direction.
5. Rain.
6. Pressure.
7. Light intensity.

For any given parameter y, the average percentage error E_y between the actual values A_x recorded by the sensors and the predicted values V_x, was computed over the whole period of measurement as follows:

$$E_y = \frac{1}{N}\sum_{t=1}^{N}\frac{|A_x(t) - V_x|}{A_x(t)} \times 100 \tag{65}$$

where N, represents the total number of values recorded and predicted.

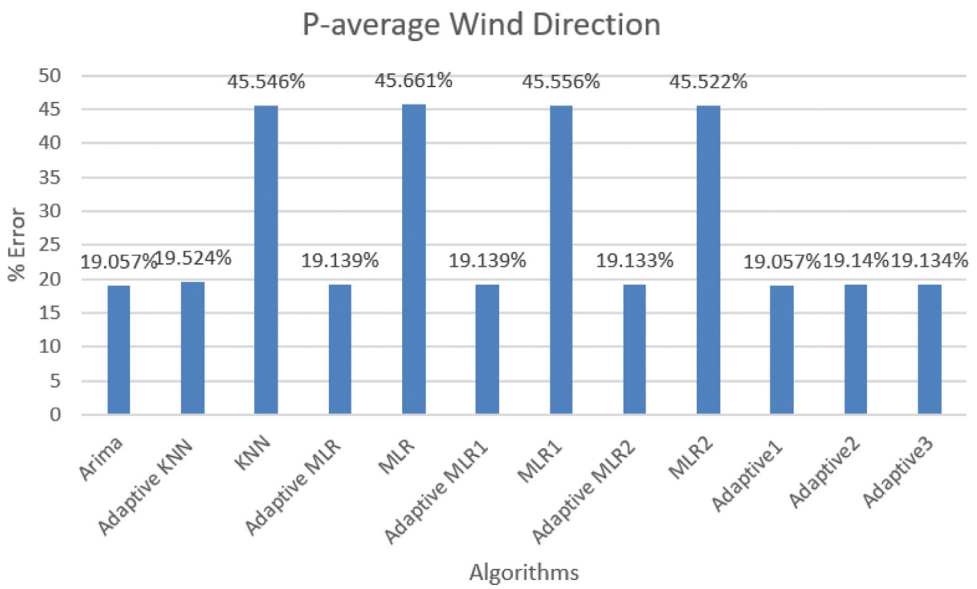

Fig. 17. Average percentage error obtained for wind direction over all three intervals with the different algorithms.

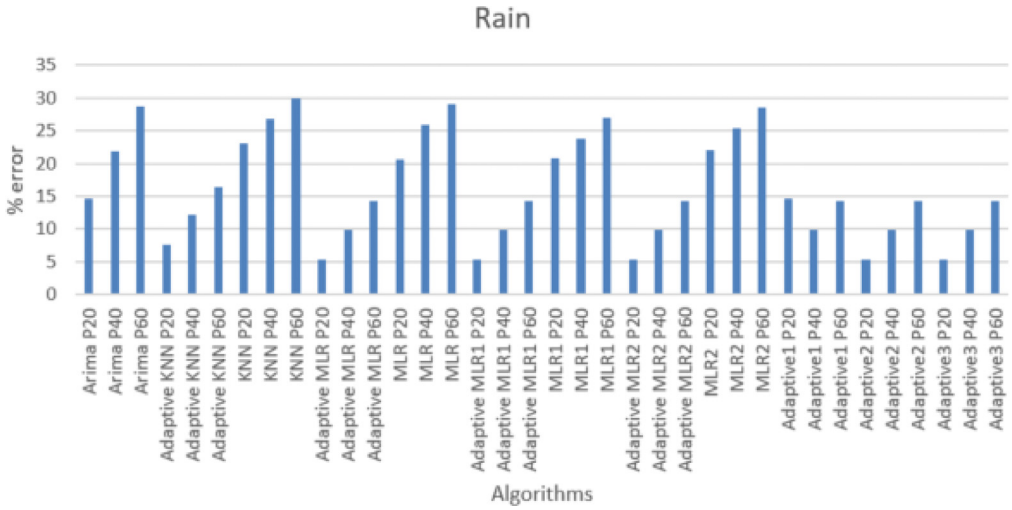

Fig. 18. Percentage error obtained for rain with the different algorithms.

The overall average % error O was also computed for all the 7 parameters as follows:

$$O = \frac{1}{7}\sum_{y=1}^{7} E_y \tag{66}$$

The Percentage error for temperature predicted by all schemes at intervals of 20 min (P20), 40 min P(40) and 60 min (P60) and are shown in Fig. 10.

The general observation is that as the prediction interval increases from 20 to 60 min, the percentage error increases from below 4% to above 6%. Moreover, the percentage error of all the adaptive schemes is lower than the non-adaptive ones. ARIMA also exhibits a low percentage error because it has an implicit adaptive function in the sense that it updates its parameters at each new registered value. Fig. 11 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average percentage error of 5.27% is achieved by the Adaptive 1 scheme. The worst average is obtained by non-adaptive MLR 2 at 6.45%.

The percentage error for humidity predicted by all schemes at intervals of 20 min (P20), 40 min P(40) and 60 min (P60) and are shown in Fig. 12.

The general observation is that as the prediction interval increases from 20 to 60 min, the % error for ARIMA increases from 6.4% to 11.4%. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Fig. 13 shows

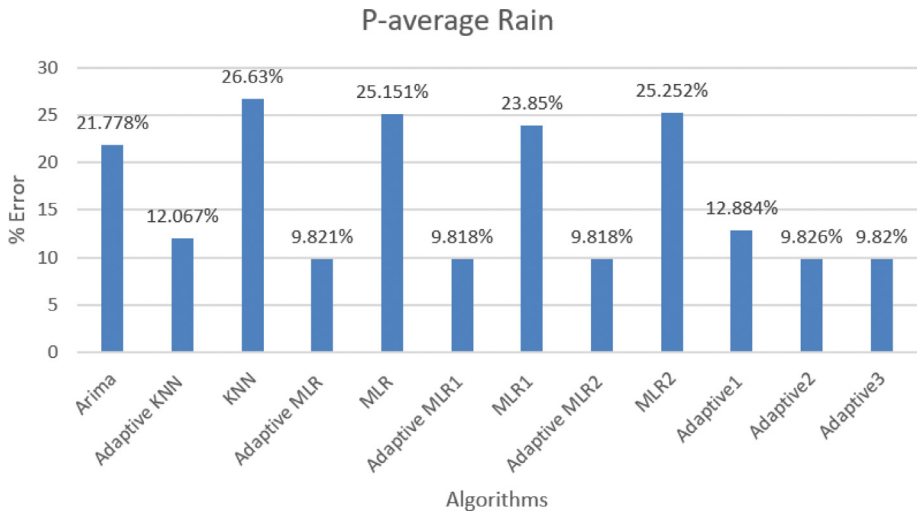

Fig. 19. Average percentage error obtained for rain over all three intervals with the different algorithms.

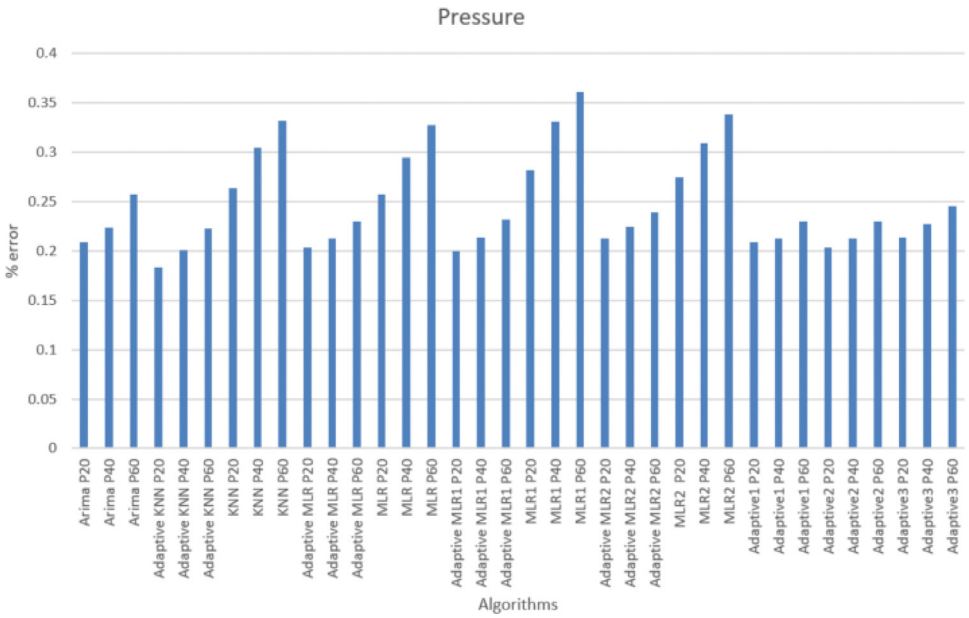

Fig. 20. Percentage error obtained for pressure with the different algorithms.

the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 9.02% is achieved by the Adaptive 1 scheme. The worst average is obtained by non-adaptive MLR at 11.23%.

The percentage error for wind speed predicted by all schemes at intervals of 20 min (P20), 40mins P(40) and 60 min (P60) and are shown in Fig. 14.

The general observation is that as the prediction interval increases from 20 to 60 min, the % error increases from 0.99% to 3.09% in the case of Adaptive MLR. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Fig. 15 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 2.038% is achieved by the Adaptive MLR, MLR1 and MLR2 schemes. The worst average is obtained by non-adaptive MLR2 at 5.767%.

The percentage error for wind direction predicted by all schemes at intervals of 20 min (P20), 40 min P(40) and 60 min (P60) and are shown in Fig. 16.

The general observation is that as the prediction interval increases from 20 to 60 min, the % error increases from 11.46% to 26.51% in the case of Adaptive K-NN. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Fig. 17 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 19.06% is achieved by the Adaptive 1 scheme. The worst average is obtained by non-adaptive MLR at 45.66%.

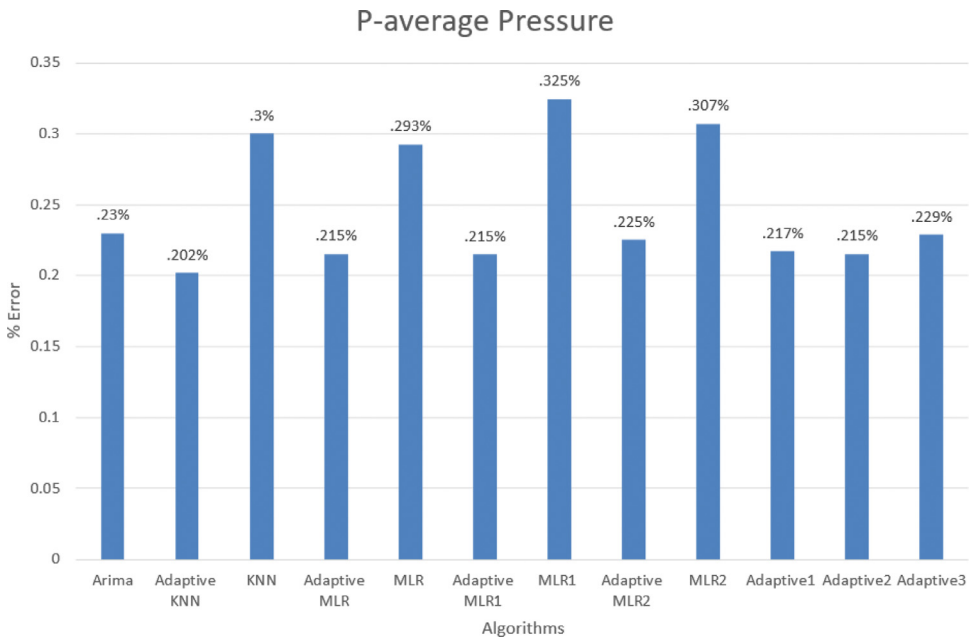

Fig. 21. Average percentage error obtained for pressure over all three intervals with the different algorithms.

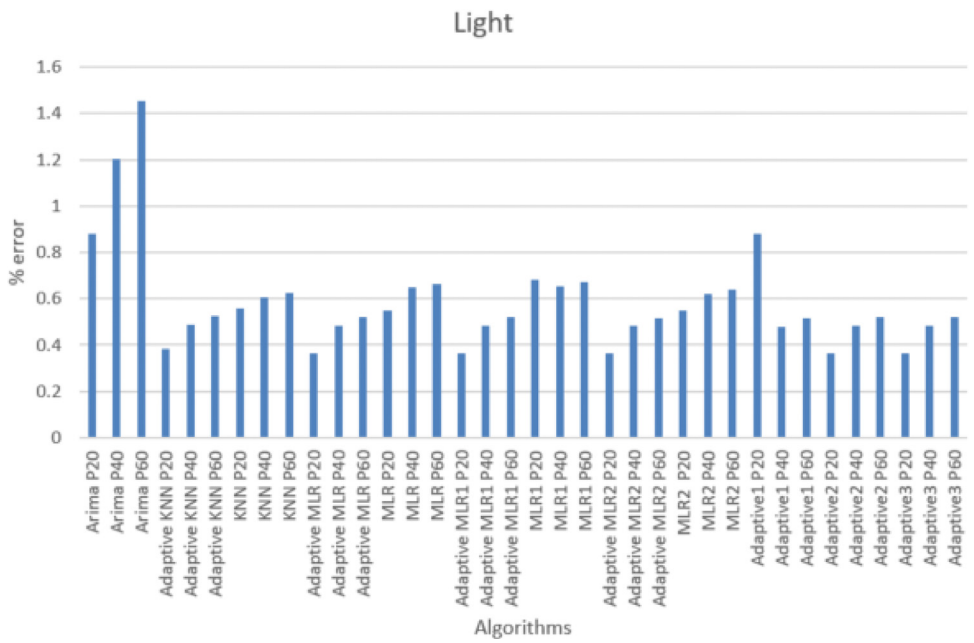

Fig. 22. Percentage error obtained for light with the different algorithms.

The percentage error for rainfall predicted by all schemes at intervals of 20 min (P20), 40 min P(40) and 60 min (P60) and are shown in Fig. 18.

The general observation is that as the prediction interval increases from 20 to 60 min, the % error increases from 5.39% to 14.22% in the case of Adaptive MLR. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Fig. 19 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 9.818% is achieved by the Adaptive MLR 1 scheme. The worst average is obtained by non-adaptive K-NN at 26.63%.

The percentage error for pressure predicted by all schemes at intervals of 20 min (P20), 40 min P(40) and 60 min (P60) and are shown in Fig. 20.

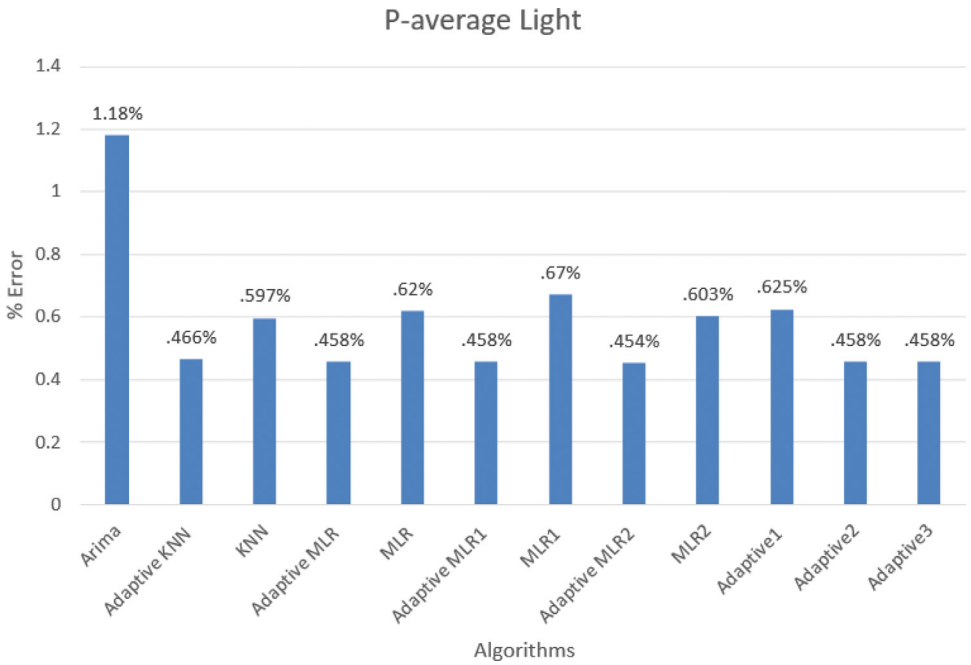

Fig. 23. Average percentage error obtained for light over all three intervals with the different algorithms.

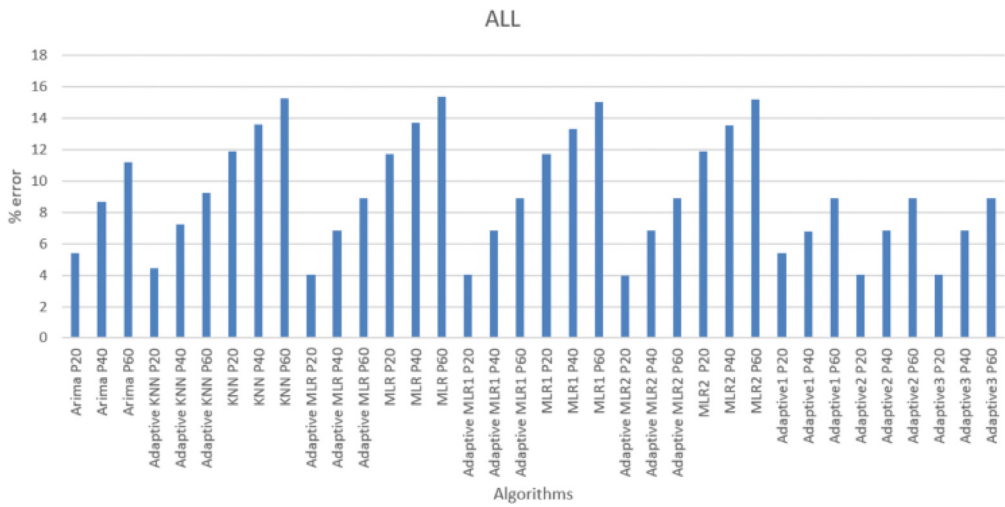

Fig. 24. Overall percentage error with the different algorithms.

The general observation is that as the prediction interval increases from 20 to 60 min, the % error increases from 0.20% to 0.23% in the case of Adaptive MLR for example. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Fig. 21 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 0.202% is achieved by the Adaptive K-NN scheme. The worst average is obtained by the non-adaptive MLR 1 scheme at 0.325%.

The percentage error for light predicted by all schemes at intervals of 20 min (P20), 40 min P(40) and 60 min (P60) and are shown in Fig. 22.

The general observation is that as the prediction interval increases from 20 to 60 min, the % error increases from 0.88% to 1.46% in the case of ARIMA. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Fig. 23 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 0.454% is achieved by the Adaptive MLR 2 scheme. The worst average is obtained by the ARIMA scheme at 1.18%.

The overall average percentage error for all parameters predicted by all schemes at intervals of 20 min (P20), 40 min P(40) and 60 min (P60) are shown in Fig. 24.

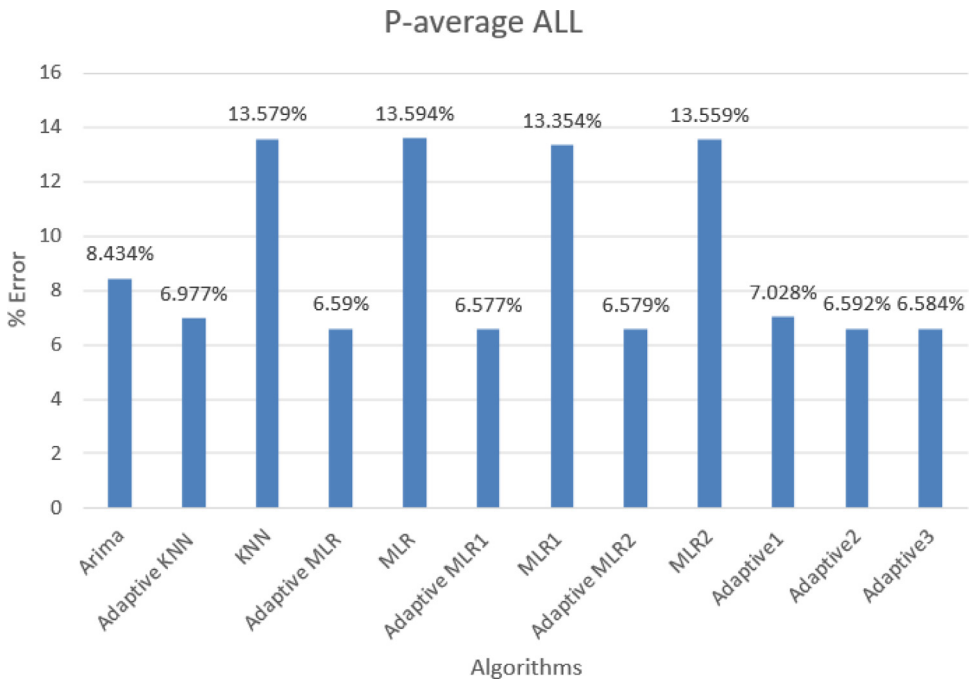

Fig. 25. Overall average percentage error over all 3-intervals with the different algorithms.

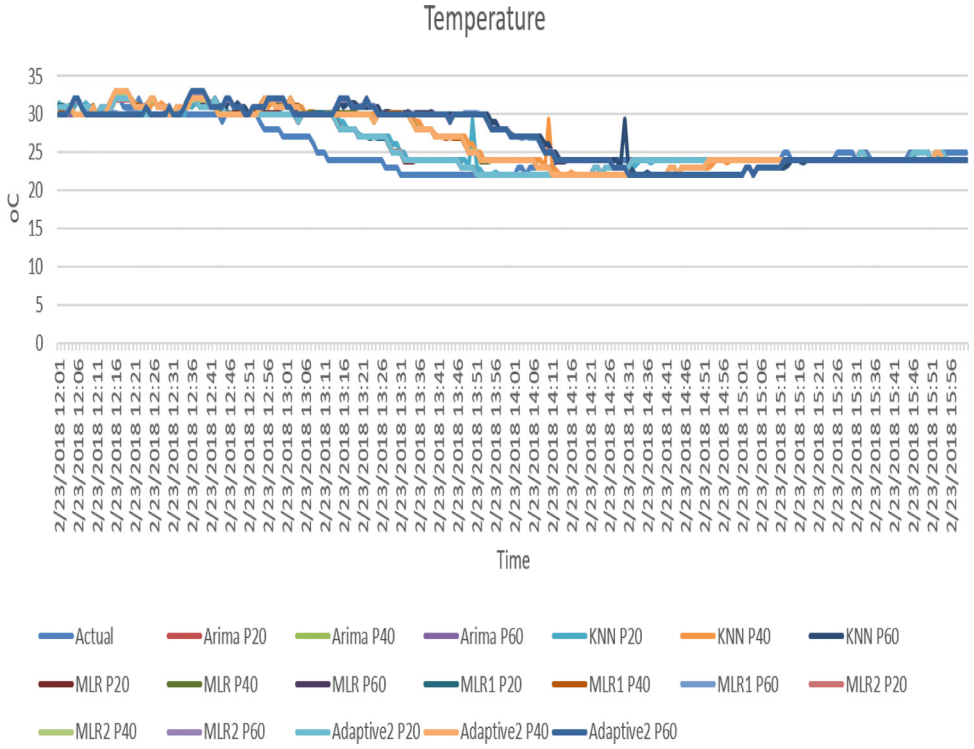

Fig. 26. Temperature readings for Day 1.

The general observation is that as the prediction interval increases from 20 to 60 min, the % error increases from 5.41% to 11.18% in the case of ARIMA. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Fig. 25 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 6.58% is achieved by the Adaptive MLR 1 scheme. The worst average is obtained by the Non-Adaptive MLR scheme at 13.59%.

The graphs in Fig. 26 show the actual and predicted temperature values for all parameters and schemes on Day 1 from 1200 to 1600. The predicted values with each algorithm used are shown in the figure. The behaviour and deviation of the predicted trends from the actual one can be observed. Similar graphs can be obtained for the other parameters: humidity, wind speed, wind direction, rain, atmospheric pressure and light intensity.

5. Conclusion

In this paper, an IoT based real-time weather forecasting system was developed and its performance analysed with both non-adaptive and adaptive prediction algorithms. The adaptive algorithms consisted of adaptive K-NN, three variants of adaptive multiple linear regression and three adaptive selection algorithms. It was observed that the lowest percentage errors for temperature, humidity and wind direction was achieved by the Adaptive 1 scheme at 5.27%, 9.02% and 19.06% respectively. The lowest percentage errors for Wind speed and Light was achieved by the Adaptive MLR 2 scheme at 2.038% and 0.454% respectively. For pressure, adaptive K-NN achieved the lowest percentage error at 0.454%. The Adaptive MLR 1 scheme achieved the lowest average percentage error for rainfall at 9.82% and for the overall average at 6.58%. The system developed is therefore accurate to predict the weather parameters investigated for short-time periods ranging from 20 min to one hour and is useful for localized forecasts especially in countries with micro-climates such as Mauritius.

Acknowledgement

The authors are thankful to the Mauritius Research Council for having funded this research work under the High Performance Computing Research and Innovation Grant (HPCRIG-A06), to the University of Mauritius and IBM Mauritius Ltd for providing the facilities required to conduct this research.

References

[1] P. Kapoor, S.S. Bedi, Weather forecasting using sliding window algorithm, ISRN Signal Process. (2013) 1–5 2013.
[2] P.H. Kulkarni, P.D. Kute, Internet of things based system for remote monitoring of weather parameters and applications, Int. J. Adv. Electron. Comput. Sci. 3 (2) (2016) 68–73.
[3] M. Burkhalter, Perle. https://www.perle.com/articles/how-iot-infrastructure-allows-for-more-accurate-weather-forecasting-40169629.shtml, 2018 [Accessed 11 August 2018].
[4] R.K. Kodali, A. Sahu, An IoT based weather information prototype using WeMos, in: 2nd International Conference on Contemporary Computing and Informatics (IC3I), Noida, India, 2016.
[5] A. Varghese, Weather based information system using IoT and cloud computing, J. Comput. Sci. Eng. 2 (6) (2015) 90–97.
[6] J.T. de Castro, G.M. Salistre, Jr, Y.-C. Byun, B.D. Gerardo, Flash flood prediction model based on multiple regression analysis for decision support system, in: Proceedings of the World Congress on Engineering and Computer Science, San Francisco, USA, 2013.
[7] M. Ancona, A. Dellacasa, G. Delzanno, A.L. Camera, I. Rellini, An "Internet of Things" vision of the flood monitoring problem, in: Fifth International Conference on Ambient Computing, Applications, Services and Technologies, Nice, France, 2015.
[8] S. Azid, B. Sharma, K. Raghuwaiya, A. Chand, S. Prasad, A. Jacquier, SMS based flood monitoring and early warning system, J. Eng. Appl. Sci. 10 (15) (2015) 6387–6391.
[9] R. Arridha, S. Sukaridhoto, D. Pramadihanto, N. Funabiki, Classification extension based on IoT-big data analytic for smart environment monitoring and analytic in real-time system, Int. J. Space-Based Situated Comput. 7 (2) (2017) 82–93.
[10] D.K. Krishnappa, D. Irwin, E. Lyons, M. Zink, CloudCast: cloud computing for short-term mobile weather forecasts, in: International Performance Computing and Communications Conference (IPCCC), Austin, TX, USA, 2012.
[11] Paras, S. Mathur, A simple weather forecasting model using mathematical regression, Ind. Res. J. Ext. Educ. 1 (2012) 161 no. Special Issue.
[12] H. Ansari, Forecasting seasonal and annual rainfall based on nonlinear modeling with gamma test in north of Iran, Int. J. Eng. Pract. Res. 2 (1) (2013) 16.
[13] S. Prabakaran, P.N. Kumar, P.S.M. Tarun, Rainfall prediction using modified linear regression, ARPN J. Eng. Appl. Sci. 12 (12) (2017) 3715.
[14] L. Li, Z. Ma, L. Liu, Y. Fan, Hadoop-based ARIMA algorithm and its application in weather forecast, Int. J. Datab. Theory Appl. 6 (5) (2013) 119.
[15] S.A. Shamsnia, N. Shahidi, A. Liaghat, A. Sarraf, S.F. Vahdat, Modeling of weather parameters using stochastic methods (ARIMA Model)(case study: abadeh region, Iran), in: International Conference on Environment and Industrial Innovation, 12, 2011, p. 282.
[16] I. Kaushik, S.M. Singh, Seasonal ARIMA model for forecasting of monthly rainfall and temperature, Environ. Res. Dev. 3 (2) (2008) 506.
[17] G.M. Datong, E.N. Goltong, An application of ARIMA models in weather forecasting: a case study of Heipang Airport – Jos Plateau, Nigeria, Innovat. Sci. Eng. 5 (2) (2017) 1–13.
[18] M. Das, S.K. Ghosh, Data-driven approaches for meteorological time series prediction: a comparative study of the state-of-the-art computational intelligence techniques, Pattern Recognit. Lett. 105 (2018) 155–164.
[19] J. Cheeneebash, A. H., A. Gopaul, Forecasting rainfall in Mauritius using seasonal autoregressive integrated moving average and artificial neural networks, Macrothink Inst. 7 (1) (2018) 115–125.
[20] M. Bannayan, G. Hoogenboom, Predicting realizations of daily weather data for climate forecasts using the non-parametric nearest-neighbour re-sampling technique, Int. J. Climatol. 28 (2008) (2007) 1357–1368.
[21] Z. Liu, Z. Zhang, Solar forecasting by K-nearest neighbors method with weather classification and physical model, in: North American Power Symposium (NAPS), Denver, CO, USA, 2016.
[22] M. Abrar, A. Mirza, Z. Jan, S. Bashir, Seasonal to inter-annual climate prediction using data mining K-NN technique, in: Wireless Networks, Information Processing and Systems. IMTIC 2008, Berlin, Heidelberg, 2008.
[23] F. Nhita, D. Saepudin, Adiwijaya, U.N. Wisesty, Comparative study of moving average on rainfall time series data for rainfall forecasting based on evolving neural network classifier, in: International Symposium on Computational and Business Intelligence (ISCBI), Bali, Indonesia, 2015.
[24] K. Abhishek, M.P. Singh, S. Ghosh, A. Anand, "Weather forecasting model using artificial neural network", 2nd International Conference on Computer, Communication, Control and Information Technology(C3IT-2012) on February 25 – 26, 2012 4 (2012) (2012) 311–318.
[25] P.T. Nastos, K.P. Moustris, I.K. Larissi, A.G. Paliatsos, Rain intensity forecast using artificial neural networks in Athens, Atmos. Res. 119 (2013) (2013) 153–160.
[26] T. Partal, H.K. Cigizoglu, E. Kahya, Daily precipitation predictions using three different wavelet neural network algorithms by meteorological data, Stoch. Environ. Res. Risk Assess. 29 (5) (2015) 1317–1329.
[27] G. Meijer, Predicting the Dutch Weather Using Recurrent Neural, University of Twente, Enschede, 2017.

[28] S. Suksri, W. Kimpan, Neural network training model for weather forecasting using fireworks algorithm, in: International Computer Science and Engineering Conference (ICSEC), Chiang Mai, Thailand, 2016.
[29] N.M. Khodabacchus, T.P. Fowdur, Performance analysis of a real-time adaptive prediction algorithm for traffic congestion, J. ICT 17 (3) (2018) 493–511.
[30] G.E.P. Box, G.M. Jenkins, G.C. Reinsel, G.M. Ljung, Time Series Analysis: Forecasting and Control, Wiley, 2015.
[31] R. Adhikari, R.K. Agrawal, An Introductory Study on Time Series Modeling and Forecasting. https://arxiv.org/ftp/arxiv/papers/1302/1302.6613.pdf, 2018 Accessed 11 May.