Mauritius Research and Innovation Council
INNOVATION FOR TECHNOLOGY

AN ADAPTIVE SHORT-TERM LOCALISED WEATHER FORECASTING SYSTEM FOR MAURITIUS

Final Report

September 2019

Mauritius Research and Innovation Council

Address:
Level 6, Ebene Heights
34, Cybercity
Ebene

Telephone: (230) 465 1235
Fax: (230) 465 1239
e-mail: contact@mric.mu
Website: www.mric.mu

This report is based on work supported by the Mauritius Research Council under award number MRC/HPC-RIG-A06. Any opinions, findings, recommendations and conclusions expressed herein are the author's and do not necessarily reflect those of the Council.

High Performance Computing Research and Innovation Grant

CONSOLIDATED FINAL Progress Report

(Updated version: 4th September 2019)

Title of Project: An Adaptive Short-term Localised Weather Forecasting System for Mauritius	**Reference No.:** MRC/HPCRIG-A06…..
Name of Local Company/Institution: University of Mauritius	**Name of Collaborating Company/Institution:** IBM Mauritius Ltd **Name of Other Collaborating Institution/s: NA**
Name of Project Leader: Assoc Prof (Dr) T.P. Fowdur **Name of Accountant/Head of Finance:** Mrs C.K. Bholah	**Name(s) of Collaborators:** Mr V.Hurbungs Dr. V. Bassoo Mrs V.R. Seetohul Dr Y.Beeharry
Project Start Date: 21/July/2017	**Projected Completion Date:** 17/May/2019
Consolidated Report for the whole project covering the period starting ___21/July/2017_______ to __17 May 2019_______.	

STATEMENT

We certify that to the best of our knowledge:

(1) the statements herein (excluding scientific hypotheses and scientific opinions) are true and complete;

(2) the text and graphics in this report as well as any accompanying publications or other documents, unless otherwise indicated, are the original work of the signatories or individuals working under their supervision and

(3) Funds allocated by the Research and Development Working Group for the project has been used for the agreed purposes of the project and according to our institutional and company regulations.

Signature of Project Leader: ……………………………………………..

Associate Professor (Dr) T.P. Fowdur

Signature of Collaborators:

…………………………	……………………..	…………………………	……………………
Mr V.Hurbungs	Dr. V. Bassoo	Mrs V.R. Seetohul	Dr Y.Beeharry

Signature of Accountant/Head of Finance: ________________________________

The progress report should follow the below format and should be sufficiently detailed to allow a proper assessment of the work undertaken during the period covered.

ACKNOWLEDGEMENTS

The investigators would like to express their sincere thanks to the following organisations and persons whose contributions have been monumental in the completion of this project:

1. The Mauritius Research Council
2. IBM Mauritius Ltd
3. The University of Mauritius
4. Members of the review panel of this project: Professor(Dr) Soodursun Jugessur, Dr Ram Dhurmea and Mr A Seegoolam
5. Mrs P Veer Ramjeawon from the MRC
6. Miss Angela Liu Kiat from IBM Mauritius Ltd
7. Miss Estelle Chan Moo Lun, Research Assistant on this project from 21 Aug 2017- to 14 May – 2018
8. Mr Jean Alberto Sunglee, student at UOM

We are indeed really obliged and grateful for the support we have received from the above organisations and individuals and look forward to continue collaborating with them on our future projects.

TABLE OF CONTENTS

EXECUTIVE SUMMARY

With the advent of global warming and other climatic imbalances, several countries are experiencing drastic weather conditions such as flash-floods which lead to major collateral damage and life loss. Predicting such weather conditions with conventional forecasting systems is not possible because these systems provide predictions for large regions over hours. Recently several real-time weather forecasting systems based on the Internet of Things have been developed to provide short-term real time forecasts also referred as Nowcasts. The main challenge in these systems is to be able to use appropriate prediction algorithms that can predict different weather parameters with the highest possible accuracy. In this project an IoT based weather forecasting system has been implemented to provide short-term weather forecasts in Mauritius at intervals ranging from 20 minutes to one hour. Several adaptive forecasting algorithms based on variants of the Multiple Linear Regression as well as the K-Nearest Neighbors (KNN) technique 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. Tests were performed in three different regions in Mauritius namely Reduit, Terre Rouge and Paillotte. A cloud notification system based on the IBM Bluemix platform was also set up to provide real-time weather predictions to users on their mobile phones. The best adaptive schemes were 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%.

CHAPTER 1
INTRODUCTION

In this chapter an overview of conventional weather forecasting as well as the climate profile and weather forecasting systems available in Mauritius is given.

1.1 Overview of Weather Forecasting

The erratic behaviour of the weather associated to effect of climate change is presently becoming an increasingly important subject of concern. A typical description of climate change is in terms of average changes in precipitation or temperature. However, it is the shifts in severity and frequency of disastrous events which result in most economic and social costs [1]. With 2005 globally marked as the warmest year since 1880 [2], the former fact is well illustrated by the several costly weather disasters in 2010. These two years were marked in history due the exceptional damaging weather events, such as the Hurricane named Katrina in 2005 and the deadly Russian heat wave in 2010. Some of the remarkable events of 2010 include: the biggest flood of Pakistan, the driest year of South-West Australia, and the warmest year in Canada. A similar trend continued in 2011. The U.S. experienced its second hottest summer throughout history, Australia went through flooding on a large scale, Texas, Arizona, and New Mexico went through shattering wildfires and drought, together with notable floods in Lower Mississippi, North Dakota and the North East [3]. Without any doubt, weather forecasting is crafting a very important and critical role to play with these unavoidable modifications in the climate of the world which are leading to catastrophic events [4].

The aim of weather forecasting is to determine in what way the weather will vary over a given interval and what will be the status of the weather over the forecast period. To achieve this, several parameters have to be measured in the atmosphere for e.g. atmospheric pressure, wind direction and speed, temperature, precipitation, cloud cover and so on. There are also computational models which can give weather forecasts based on temporal atmospheric changes caused by several factors such as warming differences across the earth's surface from solar radiation, cooling at night, atmospheric warming as a result of latent heat release during condensation, etc. Once weather data has been collected, there are several conventional techniques that are used to process these data and give a weather prediction. An overview is given next [5].

(i) Persistence forecasting is a prediction based on the assumption that the future weather condition will be an extrapolation of the present one i.e. the weather conditions will remain constant [5].

(ii) Steady-state or Trend Forecasting is a method where the forecaster observes the changes happening in the weather systems. These could be the air masses, high and low pressure systems and fronts. The forecast is based on the hypothesis that these changes will persist at the same rate they have been occurring [5].

(iii) Analogue method or pattern recognition – It uses the assumption that available weather patterns on weather charts resembling previous weather patterns on previous charts must lead to similar weather elements, or phenomena, produced by the previous patterns. It is also referred as Synoptic weather prediction. It concerns the observation of different weather elements within a given observation time. A forecasting station generates a sequence of synoptic charts every day, which are the very basis of weather prediction [6]. It consists of a large collection and analysis of weather data captured by thousands of weather stations. The images provide data for various types of atmospheric changes and expected change in weather due to current atmospheric conditions of a particular area.

(iv) Climatological Forecast - This method uses statistical records such as the mean value of weather parameters elements of an area, the highest or lowest frequency of occurrence of some weather phenomena, the maximum and minimum values of weather parameters, etc. to predict the value of those weather parameters for a given future interval [5].

(v) Numerical Weather Prediction – It uses mathematical models of the atmospheric processes that create changes to weather elements; such as, pressure, wind speed, temperature, wind direction, moisture content, etc., which have an effect on the atmospheric [5]. Numerical weather prediction (NWP) models are appropriate for large-scale medium-range weather forecasting. Typically, this "state of the atmosphere," or "picture" is defined by weather values at several disparate locations, called grid points, at ground or sea level as well as vertically in the atmosphere. Essentially it is the troposphere and lower stratosphere that are concerned. After obtaining the weather observations and the values of the measured weather elements have been input entered into the program, the equations of the model can be solved to determine new values of the weather elements for a given time interval in the future.

However, conventional weather forecasting systems provide predictions over extended periods of time for large regions which could span several Kilometres. Weather stations, generate forecasts based on a set of observations and computer models obtained from stations that could be very far from the region whose weather is being forecast. Given the large coverage area of these forecast models, a 40% probability of rain implies there is a 40% probability of rain at any given point across the whole area. In other words, the forecast is being made for a large number of individuals separated by large distances. Hence it's realistically unfeasible to obtain a correct forecast for everyone in a given TV or radio market. Realistically therefore, an 80% chance of sunshine means that there is even a possibility of rain somewhere [7]. Another limitation is that drastic changes in climatic conditions such as flash floods and cloudbursts are difficult or too costly to predict in real-time. For example Cloud Burst forecasting is predicted by the prediction of rainfall and formation of clouds. The satellite based systems are expensive and require full system support. Another technique for cloud burst prediction is Data Mining techniques for weather prediction. Data mining is a technique for extracting meaningful patterns from large amount of data. Data mining is also defined as the process of extracting implicit, previously unknown and meaningful information and knowledge from large amounts of incomplete, noisy, random and ambiguous data for practical application. Laser beam atmospheric extinction measurements from manned and unmanned aerospace vehicles are also a method to predict cloud burst. The technique consists of making measurements of the laser energy incident on target surfaces of known geometric and reflective properties, using infrared detectors or infrared cameras calibrated for radiance. It is too costly and requires full government support for deployment [8].

Given the limitations associated with conventional weather forecasting systems, several systems based on wireless sensor networks connected to micro-controllers that can even be relayed to a cloud computing facility have been used. These systems can incorporate several forecasting algorithms based on artificial neural networks, fuzzy logic, time series analysis and regression. The emergence of the Internet of Things (IoT) has also provided new avenues to address the weather forecasting problem. Essentially, these systems are able to provide localised, real-time weather forecasts. An overview of such systems is given in Section 1.2.

1.2 Climate profile in Mauritius and the existing weather forecasting system

Mauritius Island has an area of about 1844 km^2 and located in the Indian Ocean at approximately 20^{0}S and 57^{0}E on the globe as depicted in Fig. 1.1. The structure of the island is a central hill and flat terrain also known as the caldera. The general characteristics of its climate can be categorized as a pleasant and moderate tropical one. According to statistics, the usual annual rainfall is about 2120 mm with average annual temperatures of about 22^0C. The small size of the island does not make it indifferent to the significant alterations in the features of the climate such as rainfall caused by variations in the parameters: distance from coast, elevation, windward-leeward locations apart from the influences from cold fronts, irregular storms and Inter Tropical Convergence Zone (ITCZ) [9]. In contrast to other countries across the globe, there are only two seasons which are observed in Mauritius which are: summer (wet and warm) from November to April and winter (dry and cold) from May to October [10].

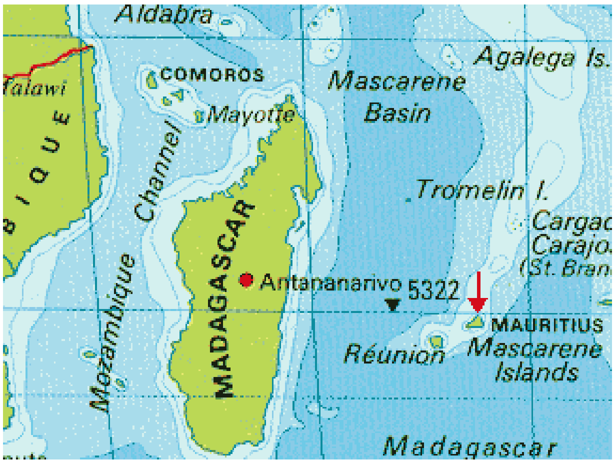


Figure 1.1: Location of Mauritius [27].

In addition to the general annual climatic characteristics of the island, each of the four regions have different specificities. For instance, the northern region experiences dry conditions from August to October while wet conditions from November to April. Approximately 70% of the average annual rainfall is accounted by these wet months. The western region experiences the result exhausting water vapour on the windward slopes and descent of air on the leeward slopes. As a matter of fact, July to September are the driest months while in the summer months November to April. 78% of the rainfall in the west occurs. The rainfall patterns of the eastern and southern parts share some similarities. They are hit by the moisture-containing oceanic air almost throughout the year and benefit from the forced uplift as a result of its passage over the sloping lands and hills [9, 10].

The three main life-threatening incidences which have openly struck the Mauritian communities are: rise in sea level; flash floods and tropical cyclone. Some other connected natural occurrences which have been witnessed are: wave surges; forest fires; coral bleaching; landslides; and droughts [11]. The location of Mauritius Island make it extremely vulnerable to intense tropical cyclones which cause blasts of wind with speeds beyond 260 km/h, along with torrential rain occurrences which frequently go above the 400 mm mark. These types of tropical cyclones are accountable for severe damages to private and public structure, loss of human lives, farming and agriculture, erosion of beaches due to wave surges amongst others. Current historical archives have shown that stronger cyclones with much longer life-span are born in the South-west Indian ocean area [12]. Over the past decades, intense tropical cyclones having a more extended diameter and average surface winds beyond 212 km/h have been witnessed. Examples of these cyclones are: Bansi and Eunice in 2015 [13, 14].

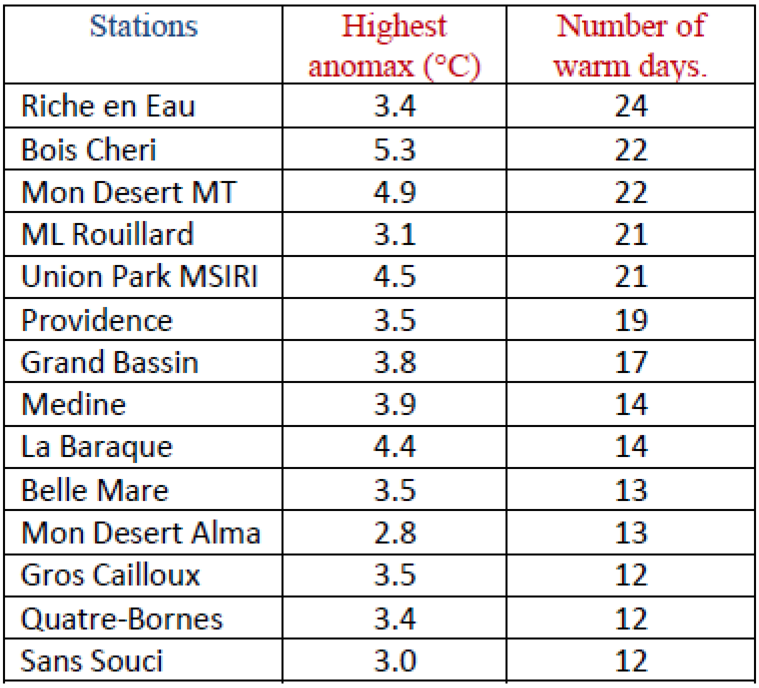
The long term annual average rainfall of 2,010 mm (measured between 1971 to 2000) is expected to decline according to the (Intergovernmental Panel on Climate Change, 2007) IPCC (2007) [15] (United Nations Environment Programme (UNEP), 2014) and Mauritius is already facing sparse distribution of rainfall from 4,000 mm on the Central Plateau to 900 mm in the western region. Previous reports of the MMS have demonstrated a rise in the frequency of droughts over the years as well as an extreme lack of rainfall in the years: 1983 to 1984; 1998 to 1999; and 2011 to 2012. Particularly, it is estimated that the mean annual rainfall will experience a degradation by 8% (Intergovernmental Panel on Climate Change, 2007) [15] with a rise in the frequency of flash floods. The 30th March 2013 marked history in Mauritius with the life-taking flash flood which occurred in the capital city of Port Louis killing 11 people with the 152 mm of rain falling in a very short lapse of time [16]. Places such as La Butte, Montagne Ory, Quatre Soeurs and Chitrakoot experience landslides during heavy rainfalls.

The rise in the level of sea is forecasted to be between 18 and 59 cm by the year 2100. The average tidal gauge records from 1950 to 2001 reveal a rise of 7.8 cm in the sea level around Mauritius and 6.7 cm around Rodrigues Island [16]. This trend would ultimately lead to erosion of the beaches, loss of bays and irreversible damages to built-up regions around the coast. The rise in sea level has stressed the influences of storm flows which are threats to the lovely littoral landscape [17].

The Mauritius Meteorological Services (MMS) is a Government Institution which operates under the Ministry of Environment with the aim of executing meteorological and other connected tasks deemed as the duty of the State by the Government. This responsibility is undertaken to support the security, safety and general well-being of the people and to accomplish the global duties under several United Nations treaties, more precisely, the World

Meteorological Organization. Table 1.1 shows the temperature anomalies recorded by the weather stations in Mauritius for the month of January 2019 [18].

Table 1.1 :Temperature anomalies recorded by the weather stations in Mauritius for the month of January 2019 [18].



Stations	Highest anomax (°C)	Number of warm days.
Riche en Eau	3.4	24
Bois Cheri	5.3	22
Mon Desert MT	4.9	22
ML Rouillard	3.1	21
Union Park MSIRI	4.5	21
Providence	3.5	19
Grand Bassin	3.8	17
Medine	3.9	14
La Baraque	4.4	14
Belle Mare	3.5	13
Mon Desert Alma	2.8	13
Gros Cailloux	3.5	12
Quatre-Bornes	3.4	12
Sans Souci	3.0	12

Figure 1.2 gives an overview of the January maximum temperature at Plaisance from the year 1969 to 2019 [18].

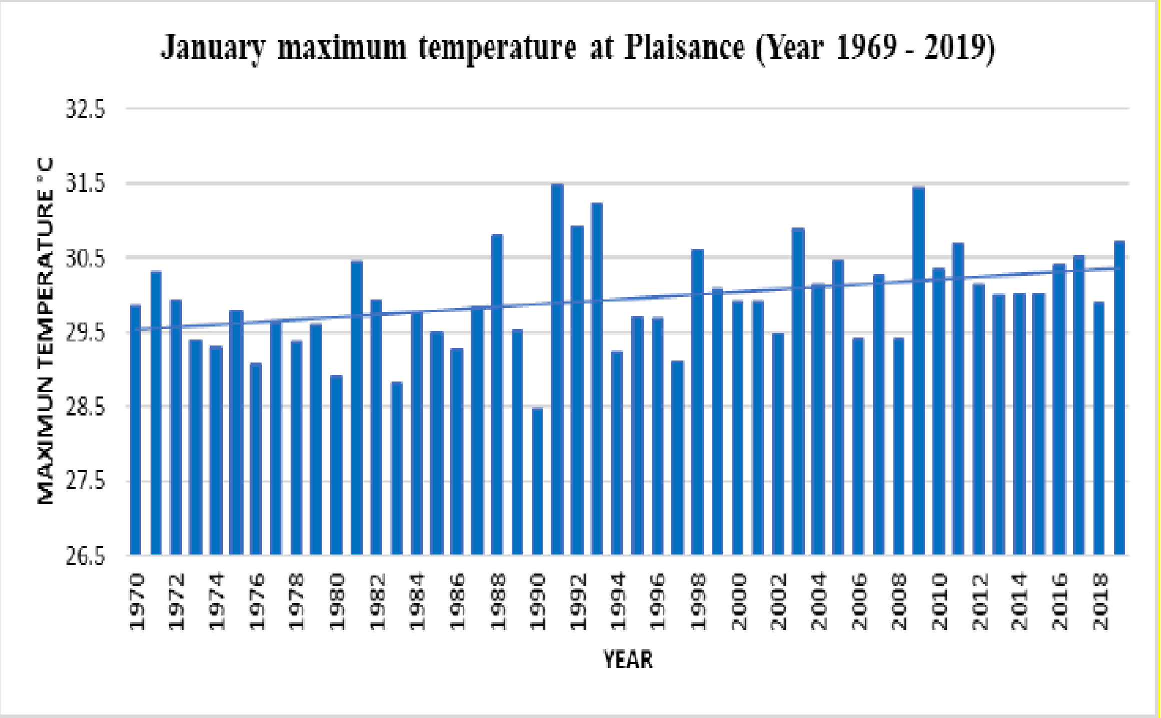


Figure 1.2: January maximum temperature at Plaisance from the year 1969 to 2019 [18].

CHAPTER 2

ANALYSIS OF WEATHER FORECASTING SYSTEMS AND ALGORITHMS

In this chapter an overview of weather forecasting systems and algorithms developed in previous research works is given. Section 2.1 gives and overview of IoT based weather forecasting systems and Section 2.2 gives an overview of numerical weather forecasting algorithms.

2.1. IoT Based Weather monitoring and forecasting systems

In [19] the authors proposed an IoT based solution for monitoring the weather at a given location and make it accessible via a cloud server. The system monitors and controls environmental conditions such as temperature, relative humidity, light intensity and CO level using sensors and relays the information to a web page. A four-tier architecture was adopted in this work in which tier 1 was used to supply information on the parameters to be monitored for noise and air pollution for the given region under control. Tier 2 consisted of appropriate sensors for measuring the environmental parameters. Tier 3 was responsible for the data acquisition from sensor devices as well as for interpreting the condition represented by the captured data parameters. The system has as its main processing unit the ATmega328 microcontroller. All the sensors and devices were attached to this main unit which could collect data from them and then transmit the data via a WiFi link to the internet. After deploying the system, the data sensed were successfully retrieved via the monitoring server. Once connected to the webpage, real-time information on intensity of sound and the CO level variations in that particular region were obtained.

In [20], the authors made use of an IoT concept. They discussed how data collected from various sensors, using an advanced microcontroller platform, can be transmitted with different communication techniques before uploading them on the internet. The proposed device has been designed to keep track of the weather parameters from faraway. An Arduino Ethernet shield has been used to connect to the internet. Tweets are sent from the Arduino with the Arduino Tweet library. For every tweet, the data are time-stamped and an incremented counter is sent along. The Arduino Tweet Library makes use of an HTTP POST. The OAuth token and the weather data are put into the URL. After deploying the system, followers of the twitter account are notified at a prearranged time to obtain details of daylight, temperature and humidity along with a counter. The data can be accessed from anywhere with any device connected to the internet.

In [21], an automated weather monitoring system has been implemented. The system automatically records humidity, temperature, light intensity, dew point and heat index by making use of three sensors, temperature and humidity sensor (DHT11), light dependant resistor and temperature sensor (LM35) , through an Arduino microcontroller. The sensors are connected to the Arduino, which is then connected to an LCD and a PC. After the system has been set up, the sensors' parameters are properly retrieved in a stable condition and kept in files. The files were then automatically imported to excel by using macros. The data were cleaned and formatted before their graphical charts were plotted. The investigation was carried out in a controlled way.

In [22], an IoT based weather station was introduced. The proposed model comprise of 4-tiers. Tier 1 is nature, tier 2 is sensor devisors, tier 3 is secure information securing and basic leadership and tier 4 is astute condition. Raspberry pi2 and environmental sensors such as DHT11, rain drop sensor module KG004, MQ2 and a buzzer, have been used to record meteorological measurements. The recorded weather parameters were sent to the cloud over a wifi connection before providing a live reporting. “Thing Speak” was used to store and retrieve data from sensor devices. Users can set alerts if weather parameters cross certain values. They can also view graphical representation of sensor values. After the system has been deployed, weather data: temperature, humidity, pressure, rain and CO levels could be viewed in a specific territory any time.

In [23], the author developed a sensor network for data acquisition and made use of a cloud computing platform for storage and processing. The architecture consist of four layers: perception layer, network layer, middleware layer, and application layer. Temperature, humidity and precipitation data are collected through sensors and smart objects. Zigbee was used to allow communications between devices. Collected data are sent to the gateway, which is responsible to collect, process and transmit the data to the application server using long range communication medium. Cloud computing resources have been used to extract important data from the information received from the gateway. They are then stored in the database. The K-medoid algorithm was implemented to analyse and retrieve patterns and knowledge from the data in the database.

In [24], the author proposed an Arduino based cloudburst predetermination system which calculates rainfall intensity in real time. The Arduino calculates the rain intensity with the data obtained from a rain gauge and a float switch. The latter observes the precipitation depth of the rainfall. A pump was used to remove water from the gauge when it is full. A servo, placed on the Arduino, allowed the board to be controlled from anywhere. Three stages of alarm were set up corresponding to three types of threshold values for the rain intensity. Alert messages are broadcast to mobile phones with the Arduino GSM shield placed on the Arduino. Simulation of water level indication was performed on LabVIEW, which is a breadboard simulator.

In [25], the authors designed a Flash Flood Warning System using SMS with advanced warning information. The system consists of a remote monitoring device, made up of an ultrasonic sensor, pulse detector, a wireless modem, a FEZ Domino board and a 6-volts battery powered by a solar panel. The device records the level and speed of water and sends the data to the server. A simulation was performed for the server application and modem, which receives sensors data and sends subscribers essential warning text, related to the increasing water level and water speed that might lead to flood, through Short Message Service (SMS). Seven days trials for the water level and velocity have been recorded. The recorded data were used as input in the regression prediction algorithm for forecasting any possibility of flood. In the system, water level and water flow data are instantly used to predict risk and the possibility of flash flood.

In [26], the authors made use of IoT to redesign previous wireless sensor network (WSN) based systems with the intention of refining the accuracy, efficiency and scalability of existing flood monitoring systems. Sensors used in an IoT system should be maintenance free, adopt ultra-low power management systems and communicate through the internet via machine to machine (M2M) technology. Wireless communication standards that can assist M2M are: the Wi-Fi technology (IEEE 802.11), the new Weightless (IEEE 802.11ah and IEEE 802.11af). The IoT system should allow easy integration of an additional device, secure interconnection of diverse information sources, persistent storage services, both predefined and customized analytical services, proper visualization interface and user application that can securely register new users. A dense network of rain gauges ensures the system's survival in a very severe weather condition and the system's accuracy.

In [27], 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. CloudCast's architecture was designed for a recently created radar sensor network by the center for Collaborative Adaptive Sensing of the Atmosphere (CASA). It can also be incorporated in other radar network like NEXRAD. CloudCast is made up of MC&C, which regulates the scanning of the radar; DiCloud, which allows users to perform investigations that can make use of an exhaustive amount of data in Amazon EC2 cloud resources ; and Nowcasting, a short term weather prediction algorithm. Based from a live experiment, CloudCast was able to perform accurate prediction with less than 2 minutes delay to deliver 15-minute Nowcast image to a mobile client.

In [28], the authors designed an SMS based Flood monitoring and Early Warning system. The system timely informs threatened population and concerned authorities of the water level when it goes beyond the predefined threshold value. Pressure sensor and other electronic components were placed on top of a metal pipe. An increase in water level would cause air to be trapped in the pipe. Pressure in the pipe is then calibrated into height of water by the Arduino. The Global System for Mobile Communications (GSM) module was used to send alert massages. Credit top-up and storing contact numbers were performed via SMS. The authors successfully measure water level through the pressure sensor and they proved that pressure and water level have a strict linear relationship.

2.2 Previous Research on Numerical Weather Forecasting Algorithms

In this section an overview of previous research on the four main categories of numerical weather forecasting algorithms namely Regression, KNN, ARIMA, Artificial Neural Networks and KNN is given.

2.2.1 Regression – based prediction

2.2.1.1 A simple weather forecasting model using mathematical regression [29]

A framework for predicting weather conditions for a specific station was proposed by [29]. Data was gathered locally and processed to obtain statistical measures of the hidden information. These include moving average, exponential moving average, rate of change, oscillator, moments and coefficients of skewness and kurtosis which were computed over certain intervals of time. The study was carried out at Pantnagar Station where weather data had been collected from April 1996 to March 1999. Around 80% of the weather data was used to train the model while the remaining 20% was used for validation. For obtaining maximum and minimum temperature approximation, data was collected for 15 weeks compared to 45 weeks for relative humidity. The whole data set is divided in two parts, the first is used to obtain multiple linear regression (MLR) equations and the remaining is used for testing the model. It was observed that it is better to use input parameters such as maximum temperature, minimum temperature and rain to estimate relative humidity compared to extracting features from its own time series. Rainfall estimation can be estimated using other parameters such as maximum temperature, minimum temperature and relative humidity. It is also possible to relate one weather parameter with other parameters.

2.2.1.2 Forecasting seasonal and annual rainfall based on nonlinear modelling with gamma test in north of Iran [30].

The focus of the study carried out by Ansari [30] was to resolve part of the problems related to the forecasting of rainfall by using a nonlinear modeling with Gamma Test (GT). The problems involved the lack of short term rainfall prediction. Despite existing mathematical techniques, fundamental concerns remain like simplicity, high accuracy, real time use in many stations of a region, and the low availability of inputs are still unsolved. This research focuses on rainfall forecasting in the northern part of Iran by using monthly rainfall data sets for the past four years. Furthermore, geographical longitude, latitude and elevation in every station are also used for forecasting. Input combinations on general statistics associated to Gamma Test were considered and the amount of data required to construct appropriate models for predicting annual, spring and summer rainfall was determined using M-test. The Gamma Test assesses the best mean squared error for a specific array of inputs that can be obtained before model construction. This approach can be applied to get the best embedding dimensions and length of data for modeling to attain a specific output. The potential of Gamma Test is used for specification of the main parameters on seasonal and annual rainfall. In addition, it uses GT-derived input data for nonlinear modeling of rainfall with Local Linear Regression (LLR) and Artificial Neural Networks (ANNs). The evaluation of nonlinear models is done in training and validation stages after the model construction. The best modeling results are obtained when input variables of monthly and seasonal rainfall are combined. The LLR models, which utilize different combinations of inputs (height, latitude, longitude, monthly/seasonal /annual rainfall), also works well in prediction.

2.2.1.3 A weighted multiple regression model to predict rainfall patterns: principal component analysis approach [31].

This study aims at developing a basic rainfall forecasting model by using Southern Oscillation Index (SOI) and Darwin Sea Level Pressures (Darwin SLP). Research was carried out for Zimbabwe over one year. Drought is common in Southern Africa and it affects many people, thus creating impoverishment, environmental harm and social privation. The main reason of drought is triggered by many natural phenomena which cause a drastic decrease in rainfall. In this study, dependent rainfall variable is expressed in terms of independent explanatory variables. Multiple linear regressions are utilized to depict a correlation between the dependent variable and the explanatory variables. This allows analyzing the results in modifying several determinants on the dependent variable. When the observations are calculated over time, the model becomes a time series regression model. Consequently, the statistical correlation can be used to forecast values of rainfall. To confirm the forecasting power of the model, all suppositions of multiple linear regression must be satisfied. To enhance the effectiveness of the model, principal component analysis is used. This approach use highly correlated features into main components that are less correlated with each other. The multiple least squares criterion weights each observation uniformly in deciding approximates of the parameters. This gives the impact of each data point over the parameters estimates and maximizes the effectiveness of the parameter estimation. Furthermore, a weighted multiple regression technique is used to adjust for heteroscedasticity in the error terms.

2.2.1.4 Rainfall prediction using modified linear regression [32].

The research uses the Linear Regression mathematical model to predict rainfall in different regions of Southern India. Previous systems were not able to predict approximate rainfall. The paper presents a revised Linear Regression model to forecast rainfall. Linear regression is applied on training data sets and the prediction is carried out by using rainfall as an independent variable and average temperature and cloud cover as dependent variables. The error percentage is calculated by subtracting the forecasted value from the actual value. The error percentage is introduced to the input training set and the last few steps repeated till there is no further increase in the error. The paper shows a mean error percentage of about 7%, which is a lower than models such as clustering and back propagation.

2.2.2 ARIMA

2.2.2.1 Hadoop-based ARIMA algorithm and its application in weather forecast [33].

The ARIMA forecasting algorithm based on Hadoop framework for weather prediction has been presented in this article. This paper uses daily average water pressure and daily average humidity data collected over the past decade to forecast the data of the coming fortnight. The error in daily average humidity does not exceed 10% and that for water pressure does not exceed 25%. It was concluded that as the prediction step increases, the error in both daily average humidity and daily average water pressure increase, therefore further enhancements are required for this model.

2.2.2.2 Short-term wind speed forecasting using ARIMA model [34].

The research in the paper uses the ARIMA model to forecast wind speed for a period of 7 days in Latvia. The gained ARIMA structure (p, d, q) was used Wind speed forecasts were carried out for periods of 2, 6, 12 and 24 hours ahead. Higher inaccuracies were noted in extended wind speed forecasts. The 24-hour prediction gave errors up to 42.4% when compared to real data. It was concluded that inaccurate results were obtained because of higher values of p and q, which led to the prediction function becoming linear.

2.2.2.3 Modeling of weather parameters using stochastic methods (ARIMA model) [35].

The paper presents the use of ARIMA model to forecast weather conditions for the Abadeh region of Iran. Weather parameters such as monthly rainfall, average temperature and relative humidity modeling and prediction using time series analysis and stochastic methods were carried out. It was concluded that the appropriate models for rainfall forecast was ARIMA (0 0 1) (1 1 1)12 model and for temperature forecast and relative humidity was ARIMA (2 1 0) (2 1 0)12 model.

2.2.2.4 Seasonal ARIMA model for forecasting of monthly rainfall and temperature [36].

The paper presents the Box-Jenkins time series seasonal ARIMA to forecast rainfall and temperature for monthly intervals for a region of India. Rainfall and temperature records from 1994 to 2006 were examined to predict the same parameters for the next five years. Data was examined using SPSS statistics computer package. Auto correlation function (ACF) and Partial Auto correlation function (PACF) of time series of temperature and rainfall were formed up to a leg interval of 72. Seasonal ARIMA models can forecast minimum and maximum temperature more precisely as statistics of models indicate. The accuracy of predictions made for rainfall by seasonal ARIMA model is less because data is irregular and contains many missing values, which increases white noise in the system. Accuracy of these predictions can be enhanced in the future by using these predicted values for missing values.

2.2.2.5 An application of ARIMA models in weather forecasting: a case study of Heipang airport – Jos plateau, Nigeria [37].

The paper presents the application the ARIMA model for weather prediction. The data used for generating the models consist of monthly records from 2011 to 2016 of average temperature, rainfall and wind velocity at Heipang airport- Jos Plateau, Nigeria. The variables used for the study are: temperature, rainfall and wind speed. To develop the ARIMA models for the variables, the first step in the analysis of data is experimental data analysis where a plot of the data is investigated. Correlogram and partial correlogram plots have been used to allow a better understanding about the stationarity of the variables. Close similarity between predicted and real data was noticed for temperature values. However, rainfall and wind speed predictions were less reliable.

2.2.3 KNN

2.2.3.1 KNN technique for analysis and prediction of temperature and humidity data [38].

The K-Nearest Neighbor (KNN) data mining approach was used to predict temperature and humidity data for a particular region in India. The algorithm uses monthly temperature and humidity data as input. Three matrices were developed for the year 2009, 2010 and 2011 with each column of the matrix representing the date, temperature and humidity on a particular day, whereas row consisted of the values. Humidity and temperature were forecasted for a period of 4 days. According to the authors, the model gave accuracy of around 90% for temperature and humidity prediction.

2.2.3.2 Weather analogue: A tool for real-time prediction of daily weather data realizations based on a modified k-nearest neighbor approach [39].

The paper presents a modified version of the k-nearest neighbor approach to predict weather data such as solar radiation, maximum and minimum temperature, and rainfall. The method presented employed the k-mean approach for k-NN showed favourable results for the prediction of the weather parameters that were part of the experiment. Rainfall prediction was also obtained even where no heavy precipitation for any given day was observed. The study showed that it was feasible to forecast even unrecorded day-to-day weather information with reasonable precision. However, it was also noted that historical weather data would greatly benefit the system and improve prediction capabilities.

2.2.3.3 Solar forecasting by K-Nearest Neighbors (K-NN) method with weather classification and physical model [40].

This study presents, a model for the prediction of weather and solar units based on the K-NN algorithm. Apart from load or price of energy prediction, other models which explain the inner mechanism of photovoltaic units are available. These mathematical models perform quite well in an optimum environment. The benefits of weather classification process has been considered in this study.

In this work, the three variants of the K-NN algorithm used are: K-NN with Gradient Descent, Distance Weighted K-NN, and Local Weighted Linear Regression. The prediction model is trained using historical datasets of solar irradiance, weather, and power generation from photovoltaic units. The primary step involves the cleansing of the gathered raw data. A classification / grouping into different pre-defined categories of weather is performed based on the weather conditions. A pre-estimation of the output power from the photovoltaic modules is done using a physical model of the solar units. These values are then compared to the actual historical outputs during the training process of the prediction model. A merging of the pre-estimated photovoltaic values with the matching forecasted residual obtained from the K-NN algorithms is performed before making predictions in the testing phase.

The three different categories of weather conditions used in this study are: cloudy, clear sky, and overcast. For each of the categories, three different K-NN based prediction algorithms are implemented. Results demonstrate that diverse models can be used for different categories of weather conditions. The basis of the grouping process is performed using the Direct Normal Irradiance (DNI). Clear sky days are categorized using the condition that the first order derivative of the DNI changes sign only once. The overcast days are categorized based on the condition that the total DNI in a day is lower than 800 W/m^2.

2.2.3.4 Seasonal to inter-annual climate prediction using Data Mining KNN technique [41].

This paper focuses on the use of K-NN with an approach of data mining for the implementation of the prediction system. The latter takes historical data as input for training a model with the ability to forecast weather conditions of targeted countries, cities or regions several months ahead. The K-NN based prediction model performs a distance measurement between the new sample from the testing set and the all the samples stored in the training set. The most common measure of distance is the Euclidean distance. Depending on the application, the predicted value is obtained by performing a voting or mean computation of the corresponding outputs from the K samples with the smallest Euclidean distances.

2.2.3.5 Predicting realizations of daily weather data for climate forecasts using the non-parametric nearest-neighbor re-sampling technique [42].

The precision of the K-NN mechanism was assessed as the primary objective of this work. This was performed using weather data from multiple sites in Georgia, collected on a daily basis for a whole year. Additionally, the calculation of the minimum required historical weather data for obtaining acceptable accuracies with the datasets of specific sites was the second objective of the work. The K-NN procedure was implemented using feature vectors consisting of minimum and maximum temperature, precipitation, and solar irradiance as input observed weather data. A moving window technique was then used to denote the differences across seasonal boundaries. In order to avoid the excessive influence of variables with high magnitudes on the neighbor selection, a normalization was also performed on the dataset.

In view to meeting the second aim of the study, the authors used data from 10 different sites and compared the best match obtained when using the whole dataset with either 5, 10, 20, or 25 years of historical data. Results demonstrate that the K-NN algorithm performs well with at least 25 years of historical training data.

2.2.4 Neural Network

2.2.4.1 Comparative Study of Moving Average on Rainfall Time Series Data For Rainfall Forecasting Based on Evolving Neural Network Classifier [43].

In this work, the Moving Average (MA) algorithm was used for smoothing time-series rainfall data. The pre-processed data is then used as input to the Evolving Neural Network (ENN) prediction model. The MA algorithms studied in this work are: Centered MA, Weighted MA, Double MA and Simple MA. The targeted forecast in this work is that of 1 month rainfall for Bangdung Regency area in Indonesia. The actual training dataset was the rainfall data for the period 1998 – 2012 retrieved from that area’s Department of Agriculture. The prediction model was then built using the ENN learning process. Tests were finally performed with smoothed data samples using all 4 MA algorithms as well as Modified Weighted MA. The results of Mean Absolute Percentage Error (MAPE) ratio obtained are as follows Simple MA(23.14%), Centered MA(15.66%), Double MA(17.56%), Weighted MA(28.55%) and Modified WMA(23.80%). The results show that the minimum MAPE value is obtained when using Centered MA. The MAPE ratio of 15.66% corresponds to an accuracy of 84.34%.

2.2.4.2 Weather forecasting model using Artificial Neural Network [44].

The authors of this work have studied the possibility to apply Artificial Neural Networks (ANN) in the analysis of weather conditions. The have aimed to implement predictive models with reliable accuracy levels together with comparing the performances of the different models by employing varying number of neurons, hidden layers, and different transfer functions. The irregular and non-linear nature of weather data trends has made ANN to turn out being an efficient technique in this case. Matlab (R2008a) with the Neural Network Fitting Tool was employed for the implementation and analysis of ANN with the weather dataset. A sample dataset with 365 records for the period 1999 – 2009 obtained from a station in Ontario was used in the study. The data set was split into 60% for the training set, 20% for the cross-validation set and the remaining 20% was used as the test set. The performance evaluation of each different predictive model was done using the Mean Squared Error (MSE). Models with a total of 20, 50, and 80 neurons were explored. A distribution of the neurons was performed over 5 and 10 hidden layers in each case.

2.2.4.3 An Efficient Weather Forecasting System using Artificial Neural Network [45].

In this research, the authors have presented a temperature prediction model based on the Neural Network algorithm. A sample dataset from Weather Underground has been used in this work. The dataset consisting of real-time observation for the period January 2009 to December 2009 has been used in the work. The main features available in the weather dataset are: Humidity in %, Temperature in Degrees Celsius, Dew Point in Degrees Celsius, Visibility in kilometers, Sea Level Pressure in hPa, Wind Speed in km/h, precipitation in centimeters, and Wind Gust Speed in km/h. Experimentations have been performed using different distributions of the number of neurons in the different number of hidden layers, and different learning rates. The different models obtained from the different combinations of the training parameters have been tested by predicting the temperature of some unseen days by the trained models. The results pertaining to the Root Mean Squared Error (RMSE) on unseen days are as follows: 02-Jan-2009 (Min: 0.0079, Max: 0.6905), 27-Aug-2009 (Min: 0.1257, Max: 0.8005), 09-Jun-2009 (Min: 0.0809, Max: 1.006) and 29-Nov-2009 (Min: 0.0336, Max: 1.2316).

2.2.4.4 Predicting the Dutch weather using recurrent neural networks [46].

This paper focuses on weather forecasts in the Netherlands where the power of neural networks is applied to weather. These new innovative techniques are used to generate weather forecasts which consist of the mean temperature for the coming days. The complexity of the model has been increased over time, from short term predictions using some data from weather stations, to predictions for temperature and weather type for the coming week based on weather maps. To be able to process the sequential gridded data format used for weather maps, a recurrent convolutional neural network is used. The model uses linear regression to make the predictions. The training cost or loss is determined using the mean squared error. After implementation of the model, the results were not deemed competitive with the current weather prediction models.

The forecasting results show that the model is not capable of predicting any extreme weather changes. The model does, however, predict the general trend in the mean temperature. Another interesting thing to note is the very unnatural bump from previous temperatures to the prediction. In two of the three plots, the temperature on the first day is wrong in an unnatural way. It suddenly bumps up before going down. This might be learned behaviour to reach a higher standard deviation. If this is the case, the addition of this extra cost becomes debatable as it forces the network to learn this kind of unwanted behaviour. There is no hard proof if this is really the case. The performance of the model is lacking because of the quality of the data in combination with the expectations. First off, the dataset was too small to train a deep neural network. This lead to overfitting. Secondly, making a prediction of the temperature up to one week ahead based on maps of land temperatures in Europe not reliably possible. Temperature is dependent on many variables which were not given to the model. This data was not easily available. The current data would probably work better for a country further away from the sea because the sea provides no information. Even then it is estimated that more variables and data would be necessary.

2.2.4.5 Neural network training model for weather forecasting using fireworks algorithm [47].

The work presented in this paper is targeted towards the utilization of the recently developed Fireworks algorithm (FWA) for training Artificial Neural Networks. The primary aim of the work is the prediction of average temperature on a daily basis with the use of weather parameters measured in Bangkok. The multilayer forward network structure used in this work consists of a single hidden layer. The training of the ANNs is performed using FWA for the determination of the proper biases and weights. Four different experiments were performed with the use of the local weather dataset in this work. The first one was made on a one year period from January 2012 to December 2012. The second one was made for the one- year period from January 2013 to December 2013. The third one was made for the two-year period from January 2012 to December 2013. The final experiment was performed for the three-year period from January 2012 to December 2014.

2.2.4.6. Forecasting Rainfall in Mauritius using Seasonal Autoregressive Integrated Moving Average and Artificial Neural Networks [48].

(Cheeneebash, et al., 2018) made a comparative study of the most recent computational intelligence (CI) techniques which have been applied for meteorological time series prediction purpose. The study takes into account artificial neural network (ANN), fuzzy logic, Bayesian network (BN) and other probabilistic models. It was found that the previous studies consider a single computational intelligence family or a single meteorological parameter and didn’t study the Bayesian network-based approaches. (Cheeneebash, et al., 2018) derived a hybrid CI technique, spatial fuzzy Bayesian network (SpaFBN), from the existing approaches. SpaFBN is a fuzzy extension of the spatial network (SpaBN), which reduces parameter uncertainty with the incorporated fuzziness. It overcomes the problem of discretized data. Moreover, it is an effective tool to predict meteorological time series data, especially humidity and precipitation rates.

2.2.4.7 Data-driven approaches for meteorological time series prediction: A comparative study of the state-of-the-art computational intelligence techniques [49]

(Das & Ghosh, 2017) studied and compared autoregressive integrated moving average (ARIMA) and the Artificial neural network (ANN) to predict rainfall in Mauritius. The amount of rainfall is considered for the period of July 2012 to July 2014. It was found that the data follows a seasonal trend. Four steps have been followed in seasonal autoregressive integrated moving average: specification, estimation, simulation and forecasting the multiplicative ARIMA model. Matlab toolbox has been used to model the neural network. The two algorithms were compared through their mean square error, mean absolute difference and mean absolute percentage difference. The ANN method was more accurate in predicting rainfall since it follows a nonlinear trend and has the ability to learn and train itself.

2.3 Theoretical Framework of Numerical Weather 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

2.3.1 Regression Models

2.3.1.1 Linear Regression [50]

The Linear Regression is a type of regression analysis method that models a relationship between a dependent variable and an independent variable using a linear equation. The model is expressed as:

$$T_{i+1} = a + b X_i$$

Where,

T_{i+1} is the predicted value on the y-axis (arrival time or congestion level),
a is the intercept on the y-axis,
b is the slope of the regression,
X_i is the independent variable (distance).

The intercept and slope of the regression are estimated using the Least Squares method described as follows.

The slope of the regression, b is calculated using the following equation:

$$B = \frac{SSxy}{SSxx}$$

Where,

SS_{xy} is the sum of the co-deviates,

SS_{xx} is the sum of the square deviates of X_i.

The intercept of the line is calculated using the Equation below.

$$A = T_{avg} - bX_{avg}$$

Where T_{avg} is the average of the dependent variable,

X_{avg} is the average of X_i.

The predicted value, T_{i+1} at any given point Xi is calculated as a+bX_i.

2.3.1.2 Non-Linear Regression [50]

The non-linear regression fits data to the function

$$y = a + b.\exp(-c.x)$$

where

a and c are the unknown parameters whose best estimates are to be obtained by the regression.

b is known and equals 8.0.

The test data was created for values of a = 3.0 and c = 0.5 and white noise, with a standard deviation of 0.5, was added to the generated y values.

2.3.1.3 Polynomial Regression [50]

The polynomial regression is a method in which a dependent variable is regressed on the powers of an independent variable. Polynomial regression is expressed as:

$$\mathrm{Ti+1} = \mathrm{a} + \beta_1 X_i + \beta_2 X^2_i + \beta_1 X^3_i + \ldots + \beta_D X^D_i$$

Where,

Each β_i , i = 1,2,.., D is the slope of regression with respect to the variable Xi,
D is the degree of the polynomial.

2.3.1.4 Multiple Linear Regression [51]

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 mins.

These coefficients were recomputed after every 120 mins 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:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

It is to be noted that the above derivations do not necessary tally with the actual physical relationships between these parameters. However, they provided the best results experimentally and therefore, were included in the application. An overview of the physical relationship between some weather parameters is given in Section….

2.3.2 ARIMA Models

2.3.2.1 Moving Average [52]

The Moving Average (MA) is a method which includes constructing a new time series based on the averages of a different subset of window size *k* of the original time series. Given a set of observations, the average is calculated by releasing the oldest observation and adding the next observation following the original subset in the series. The simple moving average used in this thesis is given by:

$$Y_{i+1} = \frac{1}{n}\sum_{k=i-n}^{i} Y_k$$

Where,

Y_{i+1} is the predicted value (speed of vehicle or congestion level),

n is the number of previous observations,

Y_k represents the value of the k previous observations.

K is the window size

2.3.2.2 Autoregressive Integrated Moving Average (ARIMA) [52]

The ARIMA is a stochastic time series model first popularised by Box and Jenkins (Box et al. 2015). An ARIMA model is a generalisation 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(Robert, 2017). The general expression of ARIMA forecasting is given below:

$$Y_{i+1} = \mu + \emptyset_1 Y_{i-1} + \dots + \emptyset_p Y_{i-p} + \Theta_1 e_{i-1} + \dots + \Theta_q e_{i-q}$$

Where,

p represents the order of autoregression,

q is the order of moving average (number of past error terms),

$\emptyset$ is the slope coefficient,

Θ is the moving average coefficient,

e is the error term

μ is a constant term

One typical technique to evaluate the order (p and q) of an ARIMA model is by using Akaike's Information Criterion (AIC) based on the probability of the data [32]. In this study, the AIC is applied to an array of predefined order (p and q) combined with an input time series data, and the model with the lowest AIC value is chosen as the predictor. Equation (2.3) gives the AIC function for a finite sample size S.

$$\mathrm{AIC_c}(p,q) = \mathrm{AIC} + \frac{2(p+q+1)(p+q+2)}{S-p-q-2}$$

Where AIC = -2 log L + 2(p + q + 1),

L is the likelihood of the data,

S is the size of time series.

2.3.3 K-Nearest Neighbors (KNN) Algorithm [52]

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

$$Y_{i+1} = \frac{1}{K}\sum_{i=1}^{K} y_k$$

Where,

K is the neighbourhood size,

And y_k *is* the nearest reading

Figure 2.1 illustrates an example of the KNN technique where the green dot represents the average of the most adjacent observations.

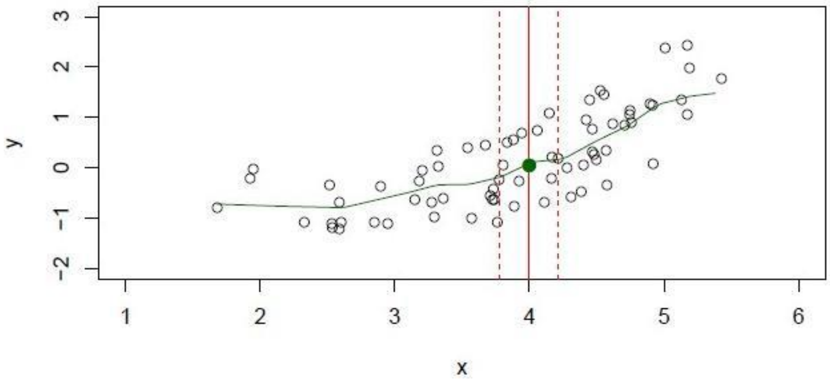


Figure 2.1: KNN Neighborhood [52]

The distance function also known as a distance metric is used to compute the distance between a pair of readings in a data set. In this work, the Euclidean Distance metric is included into the KNN technique, and the formula is given by:

$$Euclidean\ Distance\ Metric = \sqrt{(x_j - x_k\ )^2}$$

Where,

x_j is the next the interval to predict (distance),
x_k is the actual reading (distance)

Once all the distances between the unknown observation and all the real observations are calculated, the K observations with the smallest distances are chosen for prediction. The Neighbourhood size, K is radically linked to the prediction performance. Several techniques like trial-and-error and cross-validation methods have been used to determine the best neighbour size of the KNN algorithm, but require an amount of computational time. Therefore in this research, an empirical rule-of-thumb is selected where the size of K is equal to the square root of the dimension of the actual data set [52].

2.3.4 Neural Network (NN) Model [53]

The NN consists of several interconnected processing elements called neurons which are arranged in three layers namely the input, hidden and an output layer. Each neuron evolves a weight value that relates the input and output data. Multilayer Perceptron (MLP) NN architecture is used in this research due to its capability of estimating any function [54]. Figure 2.2 exemplifies this architecture.

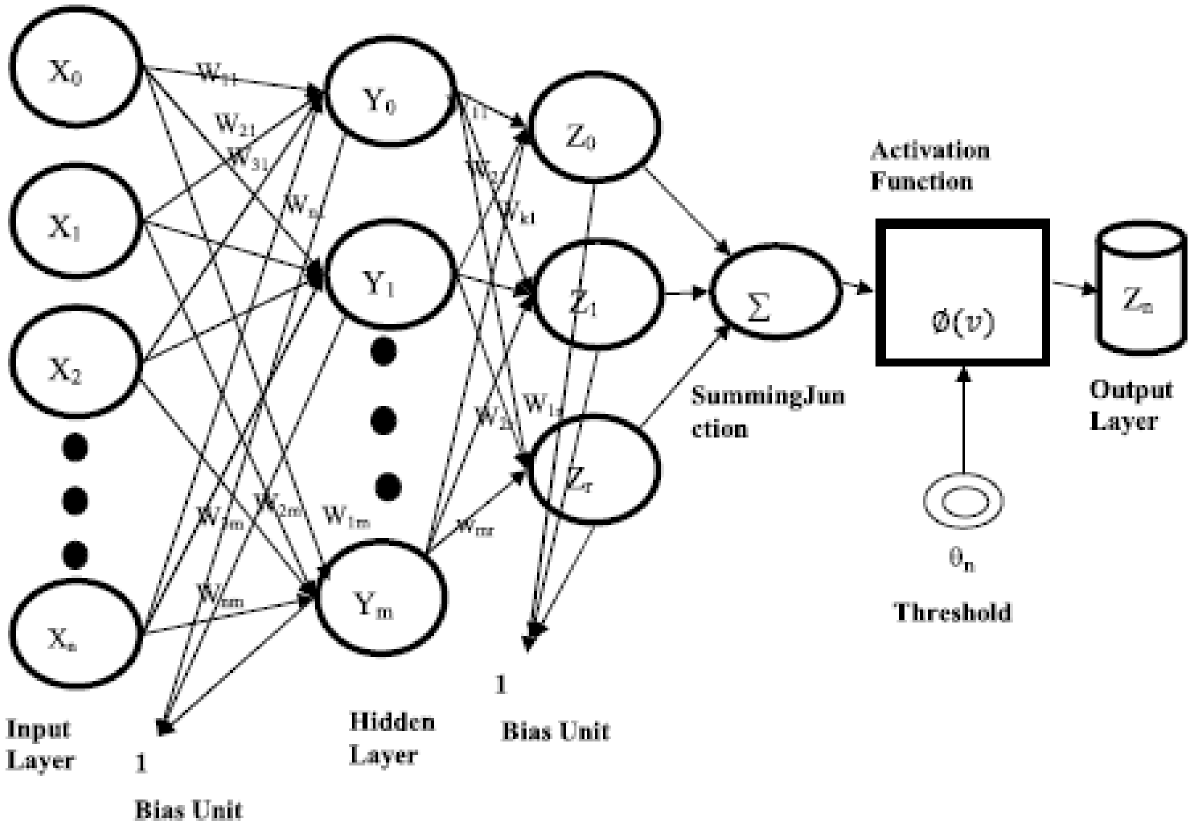


Figure 2.2: Structure of Neural Network [53]

The output of the NN is calculated as follows.

$$A = f\left(\sum_{i=1}^{N} w_i\, p_i + b_0\right)$$

Where,

A is the output value,
f is the activation function,
w_i is the weight of the ith neuron,
p_i is the value of the ith input,
b_0 is the bias value and
N is the number of neuron

The number of nodes in the input layer is proportional to the number of input variables that are used to predict an output. The hidden layer describes the complexity of the connection between the input and output layer. The number of nodes in the hidden layer is usually determined by a trial and error method since it relies on the complexity of the problem. Connections between the layers are linked with weights. Every input to the network is passed to down to the hidden layer which uses the inputs to compute a new value depending on the weights from the link between the input and hidden layer. The output layer then uses this new value to create a modified value based on the linked weight between the hidden and output layer. The output value is then used in an activation function to calculate a forecast value.

The activation function is used in the hidden layer to initiate non-linearity into the network to normalise the range of the output of a neuron to -1 and 1. The sigmoid function is applied in this research since it can adapt a range of behaviours depending on the input value. The expression of the sigmoid activation function is given in the following equation.

$$\varphi(v) = \frac{1}{1+ e^{-v}} \tag{23}$$

Where v is the input value.

The Back-Propagation algorithm developed by Rumelhart et al.[55], is applied during the learning phase of the Neural Network to balance the weights of the connection. In this research, this training algorithm is used in the implementation of the NN. The algorithm can be divided in four stages as listed below.

1. Initialises weights to random values
2. Performing feed-forward process that presents data to the network
3. Computing the error and then back propagates the error to previous layers
4. Adapt the weights of the connections.

The error between the network output and the desired output can be reduced if the above steps are repeated a number of times. The learning phase of the network is regulated by several parameters namely the Learning Rate, Momentum and Epoch [56] which are set by the user during network setup. In addition to that, it is important to normalise the input data before the training process to enhance the computations and to achieve more accurate results [57]. Hence, a mere min-max normalisation is used in thesis and is given by:

$$X_{Normalized} = \frac{X_{current} - X_{Min}}{X_{Max} - X_{Min}} \qquad [24]$$

Where,

$X_{current}$ is the current value,

X_{Min} is the minimum value of X and

X_{Max} is the maximum value of X.

2.4 Implementation of different forecasting algorithms.

The following subsections provide comprehensive information about the proposed program structure and the design of the user interface.

2.4.1 Program Design

Figure 2.3 demonstrates a binary tree organisation of the Java program's methods developed for the application of weather forecasting.

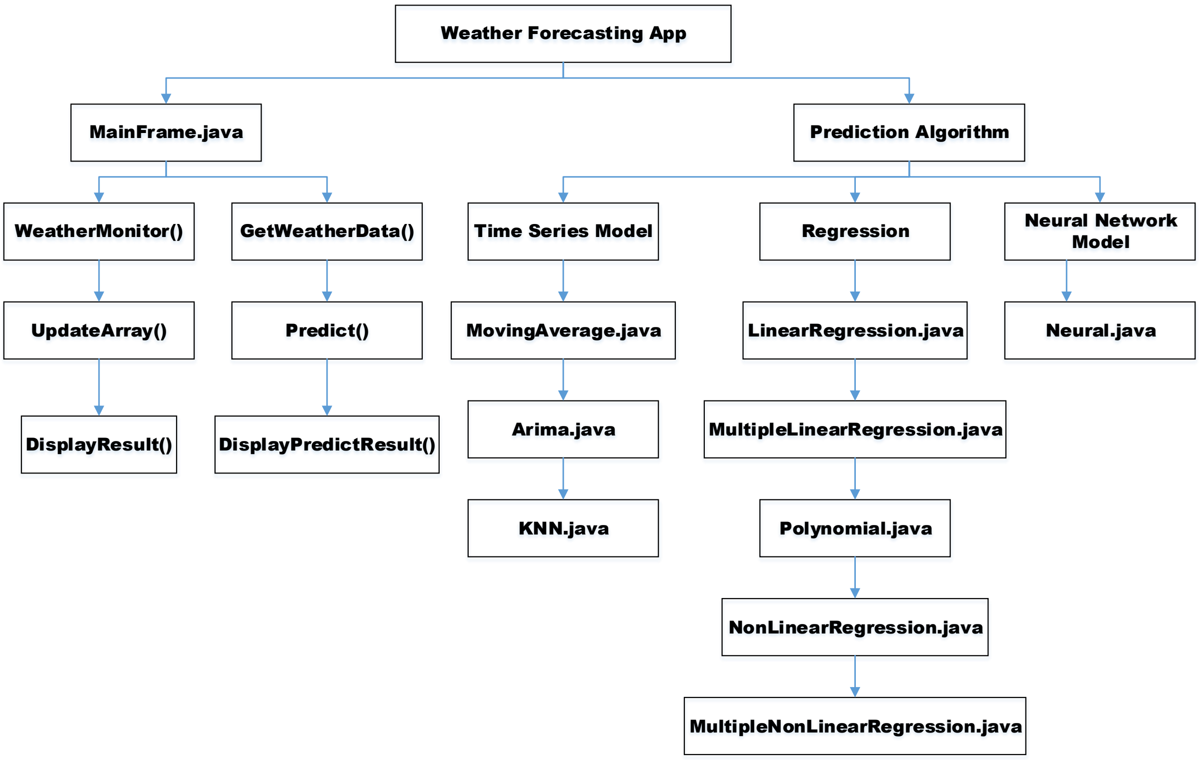


Figure 2.3: Program method structure

The application consists of eight Java classes which are made up of one Main Frame class and nine predictive analysis classes. When the program begins, the Main Frame (MainFrame.java) runs the graphical components to display the Java form given in Figure 1. 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 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.

2.4.2 Graphical User Interface (GUI)

Figure 2.4 shows the proposed graphical user interface for the weather forecasting application.

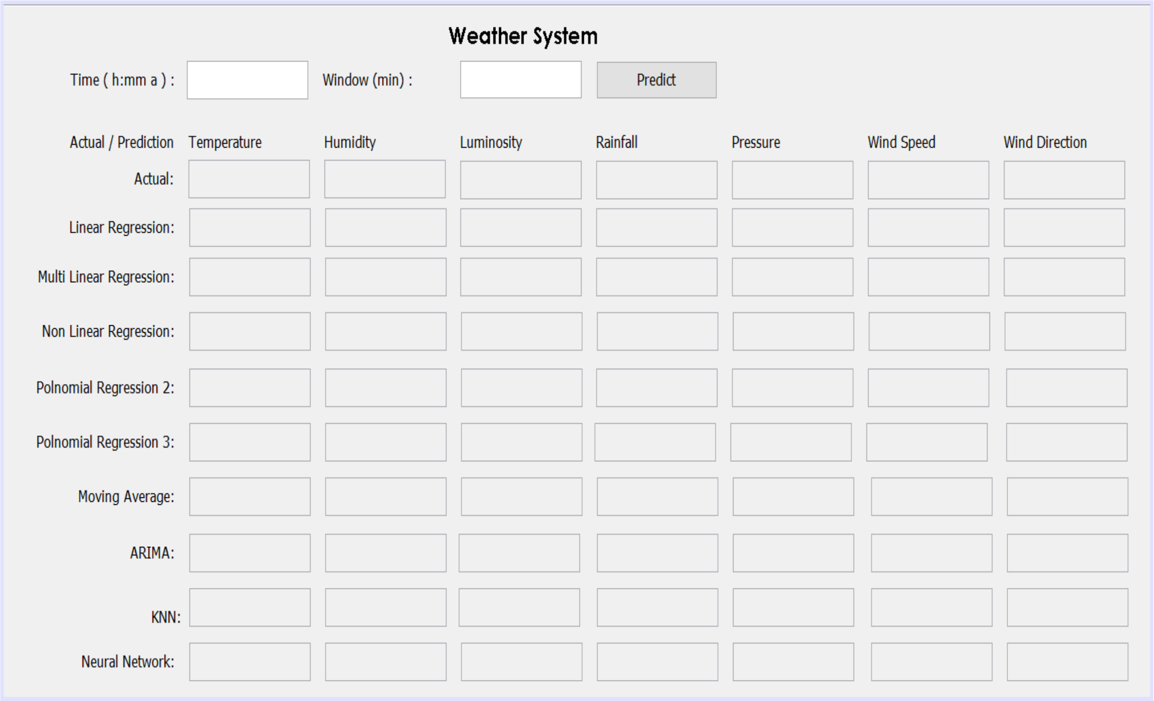


Figure 2.4: GUI

The user interface shows the actual weather data (temperature, humidity, luminosity, rainfall, pressure, wind speed and wind direction) in real time. The user data can set the time and window size for predicting weather. This user interface will then display all predicted values calculated by using the following prediction algorithms: linear regression, multi linear regression, nonlinear regression, polynomial regression with degree 2 and 3, moving average, ARIMA, KNN and neural network.

2.4.3 Implementation of Linear Regression

The linear regression class will train the model in its constructor. The constructor uses the time array, predict array (i.e. specific weather sensor data) and array size as input parameters. The slope of regression and the intercept of the line are calculated using the least squares method. A function "predict" returns the predicted value computed at a given time. The linear regression algorithm is explained in pseudocode form in Figures 2.5 and 2.6.

```
function LinearRegression (float [] y, float [] x, int size)
Initialize variable sumx, sumy, meanx, meany
Compute slope of regression and intercept
end
```

Figure 2.5: Linear regression function

```
function predict (double time)
prediction = $a + bX$ where $X$ = time
return prediction to calling method
end
```

Figure 2.6: Predict function

2.4.4 Implementation of Non-linear regression

The non-linear regression class will train the model in its constructor. The constructor uses the time array and predict array (i.e. specific weather sensor data) as input parameters. A function "predict" returns the predicted value calculated at a given time. The non-linear regression algorithm is described in pseudocode form in Figures 2.7 and 2.8. The Flanagan library has been used in the implementation of this regression.

```
function NonLinearRegression (float [] y, float [] x)
Create instances of the class holding the function, $y = a + b.exp(-c.x)$
assign value to constant b in the function
initial estimates of a and c in $y = a + b.exp(-c.x)$
initial step sizes for a and c in $y = a + b.exp(-c.x)$
create an instance of Regression
b is fixed and best estimates are evaluated for a and c
end
```

Figure 2.7: Non-Linear regression function

```
function predict (double time)
prediction = $a + b.exp(-c.x)$ , where x = time
return prediction to calling method
end
```

Figure 2.8: Predict function

2.4.5 Implementation of Polynomial regression

The polynomial regression class will train the model in its constructor. The constructor uses the time array, predict array (i.e. specific weather sensor data) and degree as input parameters. In this project, degree one and degree two have been implemented. A function “predict” returns the predicted value computed at a given time. Polynomial Regression is expalined in pseudocode form given in Figures 2.9 and 2.10. It is based on the same prediction technique and least square method as the linear regression but with an input parameter namely, degree.

```
function PolynomialRegression  (float [] y, float [] x, int degree)
Initialise residuals variables for least square method
Set degree and create matrix from input arrays
Compute mean
Find intercept and slope of regression using least square model
end
```

Figure 2.9: Polynomial regression pseudocode

```
function predict (double time)
Predict y corresponding to x, where x = time
return prediction to calling method
end
```

Figure 2.10: Predict function

2.4.5 Implementation of Multiple linear regression

The multiple linear regression class will train the model in its constructor. The constructor uses the time array and predict array (i.e. several weather sensor data) as input parameters. A function "predict" returns the predicted value computed at a given time. Multiple linear Regression is described in pseudocode form given in Figures 2.11 and 2.12. This algorithm applies all previous sensor values to forecast the next specific sensor value.

```
function MultipleLinearRegression  (float [][] x, float [] y)
Perform QR Decomposition with array x
find least squares solution
end
```

Figure 2.11: Multiple linear regression function

```
function predict (double time)
Predict y corresponding to x, where x = time
return prediction to calling method
end
```

Figure 2.12: Predict function

2.4.6 Implementation of Moving Average

The time series forecasting implements model that make use of previously recorded data to forecast future values. The time series models selected in this work are the Moving Average and ARIMA. In the main class, a function will recursively call the class MovingAverage, whereby its constructor takes the predict array (i.e. several weather sensor data) as input parameter to perform an average, and the function predict. A function “predict” returns the predicted value computed at a given time. The pseudocode of the Moving Average algorithm is given in Figures 2.13 and 2.14.

```
function MovingAverage  (float [][] values)
Perform Average
end
```

Figure 2.13: Moving Average function

```
function predict (double time)
return average from Moving Average function
end
```

Figure 2.14: Predict function

2.4.7 Implementation of Autoregressive Integrated Moving Average (ARIMA)

ARIMA function formulates an autoregression model to predict a value. The constructor takes the predict array (i.e. several weather sensor data) as input parameter. A function “predict” returns the predicted value calculated at a given time. ARIMA is explained in pseudocode form given in Figures 2.15 and 2.16.

```
function ARIMA  (float [][] values)
Initialise the variable
Create arima instance from ARMA class
Get arima model list
arima instance set its dataset = values
end
```

Figure 2.15: ARIMA function

```
function predict (double time)
Predict the next value with arima model
return prediction to calling method
end
```

Figure 2.16: Predict function

2.4.8 Implementation of K-Nearest neighbour (KNN)

The K-Nearest neighbour has an output value equals to the average of its K-nearest neighbours. The KNN constructor takes all recent data values, for a particular weather parameter and value K as input parameter before performing prediction. KNN is described in pseudocode form given in Figure 2.17.

```
function KNN  (double [][] data_all, double[] to_predict, int k )
Initialise the variables
Perform Euclidean distance for all weather parameters
Find K-nearest weather parameters
Perform average of K- nearest weather parameters
Return average
end
```

Figure 2.17: KNN function

2.4.9 Implementation of Neural Network (NN)

The Neuroph famework [58] has been used to implement the NN algorithm. Neuroph is a lightweight Java neural network framework to develop common neural network architectures. It contains well designed, open source java library with classes which match up to basic neural network concept. Both its Java neural network library as shown in Fig 2.18 and GUI tool as shown in Fig 2.19 allow the creation, training and saving of neural networks.

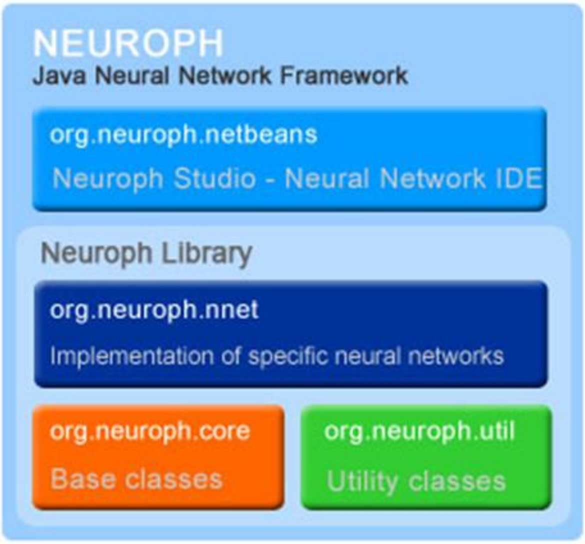


Fig 2.18 Neuroph Java neural network library [58].

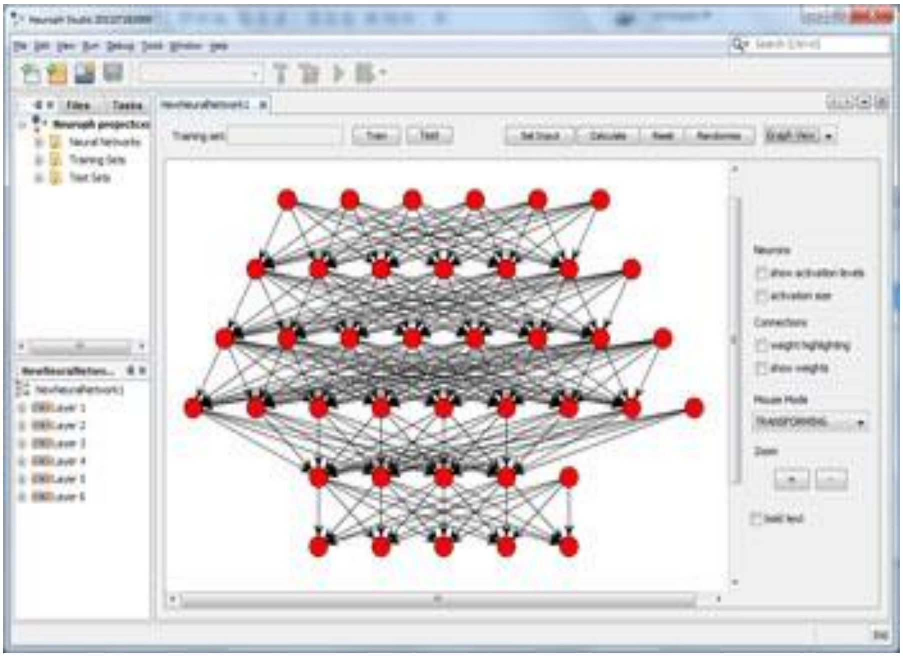


Fig 2.19 Neuroph GUI tool [58].

Neuroph supports the following neural network architectures:

1. Adaline
2. Perception
3. Multi Layer Perceptron with Backpropagation, Momentum on Resilient Propagation
4. Hopfield network
5. Bidirectional Associative Memory
6. Kohonen network
7. Hebbian network

8. Maxnet
9. Competitive network
10. Instar
11. Outstar
12. RBF network
13. Neuro Fuzzy Reasoner

In this project, Multi Layer Perceptron with Momentum on Resilient Propagation was used.

2.5 Developing an adaptive forecasting algorithm.

In this section the adaptive algorithms developed are described. Two approaches have been used to develop adaptive algorithms. The first one consists of modifying the existing Multiple Linear Regression and KNN algorithms to make them adaptive in the sense that their training windows are dynamically adjusted ever t seconds as new parameters are recorded. The second phase consists of devising a set of criteria to select the best prediction algorithm for the next prediction.

2.5.1 Adaptive Multiple Linear Regression and KNN

In order to make the MLR, MLR1 and MLR2 algorithms adaptive, the training window W, used to compute their coefficients a0, a1 and a2 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 KNN

For adaptive KNN also the training window is changed every t seconds and the modified equation is as follows:

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

2.5.2 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 KNN
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 KNN and so on.

Consider Fig 2.20 and notations that will be used in formulating the prediction algorithms:

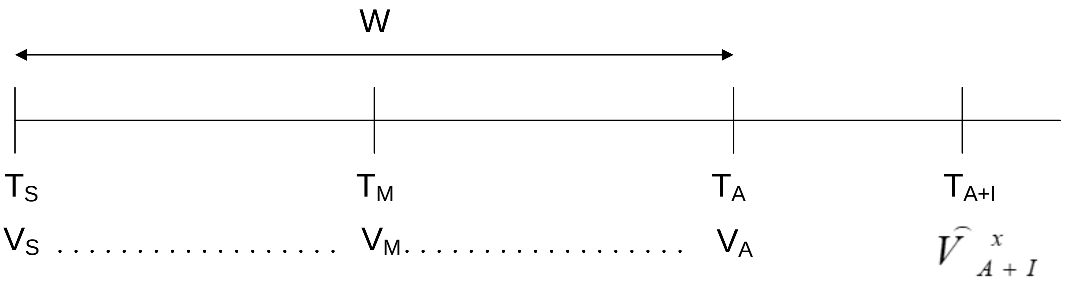

W
T_S
T_M
T_A
T_{A+I}
V_S . . . .
V_M. . . . .
V_A
$\hat{V}^{x}_{A+I}$

Figure 2.20: Timing diagram for Algorithm A1.

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.

T_{A+I}: The time at which the values of the parameter is to be predicted. I here 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.

$\widehat{V}_{A+I}^{x}$: 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:

$$\widehat{V}_A^x = P^x\left(V_S, V_{A-1}\right) \tag{48}$$

Where, $\widehat{V}_A^x$ 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:

$$\widehat{V}_A^1 = P^1\left(V_S, V_{A-1}\right)\ \widehat{V}_A^2 = P^2\left(V_S, V_{A-1}\right)\ \widehat{V}_A^3 = P^3\left(V_S, V_{A-1}\right)\ \widehat{V}_A^4 = P^4\left(V_S, V_{A-1}\right)\ \widehat{V}_A^5 = P^5\left(V_S, V_{A-1}\right) \tag{50}$$

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

$$SE_A^x = \left(V_A - \widehat{V}_A^x\right)^2 \tag{51}$$

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

$$SE_A^1 = \left(V_A - \widehat{V}_A^1\right)^2\ SE_A^2 = \left(V_A - \widehat{V}_A^2\right)^2\ SE_A^3 = \left(V_A - \widehat{V}_A^3\right)^2\ SE_A^4 = \left(V_A - \widehat{V}_A^4\right)^2\ SE_A^5 = \left(V_A - \widehat{V}_A^5\right)^2 \tag{52}$$

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:

$$\begin{aligned} &\hat{V}^x_{A+I} = P^x(V_S, V_A) \\ &x = Index\left(\min\left[SE^x_A\right]\right) = Index\left(\min\left[SE^1_A, SE^2_A, SE^3_A, SE^4_A, SE^5_A,\right]\right) \end{aligned} \tag{53}$$

Algorithm A2 is formulated as follows:

Consider Figure 2.21 which is the timing diagram for algorithm A2.

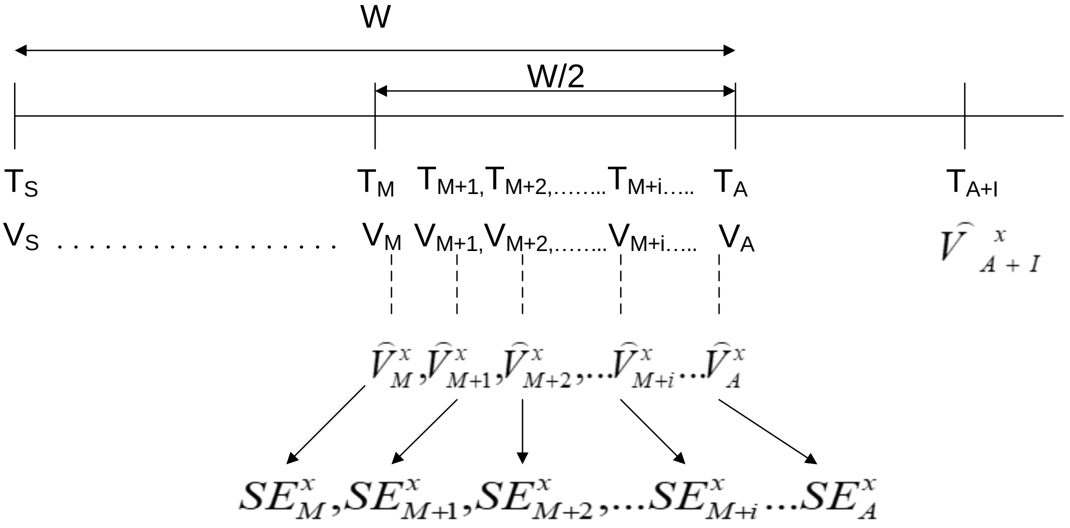

W
W/2
T_S
T_M $T_{M+1}, T_{M+2}, \ldots\ldots T_{M+i} \ldots\ldots$ T_A
T_{A+I}
V_S . . . .
V_M $V_{M+1}, V_{M+2}, \ldots\ldots V_{M+i} \ldots\ldots$ V_A
$\hat{V}^x_{A+I}$
$\hat{V}^x_M, \hat{V}^x_{M+1}, \hat{V}^x_{M+2}, \ldots \hat{V}^x_{M+i} \ldots \hat{V}^x_A$
$SE^x_M, SE^x_{M+1}, SE^x_{M+2}, \ldots SE^x_{M+i} \ldots SE^x_A$

Figure 2.21: 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 VM to VA.

$\hat{V}^x_M$ 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.

The algorithm proceeds as follows:

1. Predict the values from time T$_M$ to T$_A$ i.e. $\widehat{V}_M^x,\widehat{V}_{M+1}^x,\widehat{V}_{M+2}^x,...\widehat{V}_{M+i}^x...\widehat{V}_A^x$ at a one minute intervals with each predictor P^x as follows:

$$\widehat{V}_M^x = P^x\left(V_S, V_{M-1}\right) \tag{54}$$

More explicitly:

$$\widehat{V}_M^1 = P^1\left(V_S, V_{M-1}\right), \widehat{V}_M^3 = P^3\left(V_S, V_{M-1}\right), \widehat{V}_M^4 = P^4\left(V_S, V_{M-1}\right), \widehat{V}_M^5 = P^5\left(V_S, V_{M-1}\right) \tag{55}$$

In general:

$\widehat{V}_{M+i}^x = P^x\left(V_S, V_{M+i-1}\right)$ and:

$$\widehat{V}_{M+i}^1 = P^1\left(V_S, V_{M+i-1}\right) \widehat{V}_{M+i}^2 = P^2\left(V_S, V_{M+i-1}\right) \widehat{V}_{M+i}^3 = P^3\left(V_S, V_{M+i-1}\right) \widehat{V}_{M+i}^4 = P^4\left(V_S, V_{M+i-1}\right)$$

$$\widehat{V}_{M+i}^5 = P^5\left(V_S, V_{M+i-1}\right) \tag{56}$$

2. Compute the squared Euclidean distances $SE_M^x, SE_{M+1}^x, SE_{M+2}^x, ...SE_{M+i}^x...SE_A^x$ between the actual values V$_M$,V$_{M+1}$,V$_{M+2}$,….V$_{M+i}$…..V$_A$ and $\widehat{V}_M^x,\widehat{V}_{M+1}^x,\widehat{V}_{M+2}^x,...\widehat{V}_{M+i}^x...\widehat{V}_A^x$ respectively for each algorithm as follows:

$$SE_M^x = \left(V_M - \widehat{V}_M^x\right)^2 \tag{57}$$

More explicitly:

$$SE_M^1 = \left(V_M - \widehat{V}_M^1\right)^2 \; SE_M^2 = \left(V_M - \widehat{V}_M^2\right)^2 \; SE_M^3 = \left(V_M - \widehat{V}_M^3\right)^2 \; SE_M^4 = \left(V_M - \widehat{V}_M^4\right)^2 \; SE_M^5 = \left(V_M - \widehat{V}_M^5\right)^2$$

(58)

In general:

$SE_{M+i}^x = \left(V_{M+i} - \widehat{V}_{M+i}^x\right)^2$ and more explicitly:

$$SE_{M+i}^1 = \left(V_{M+i} - \widehat{V}_{M+i}^1\right)^2 \; SE_{M+i}^2 = \left(V_{M+i} - \widehat{V}_{M+i}^2\right)^2 \; SE_{M+i}^3 = \left(V_{M+i} - \widehat{V}_{M+i}^3\right)^2 \; SE_{M+i}^4 = \left(V_{M+i} - \widehat{V}_{M+i}^4\right)^2$$

$$SE_{M+i}^5 = \left(V_{M+i} - \widehat{V}_{M+i}^5\right)^2 \tag{59}$$

3. Compute the mean squared error for each predictor for all predictions made from T$_M$ to T$_A$ of the values $\widehat{V}_M^x,\widehat{V}_{M+1}^x,\widehat{V}_{M+2}^x,...\widehat{V}_{M+i}^x...\widehat{V}_A^x$. For a given predictor Px taking i from 1 to W/2, we obtain the MSE as follows:

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

More explicitly:

$$\overline{MSE}^1 = \frac{2}{W}\sum_{i=1}^{W/2} SE_{M+1-i}^1 \quad \overline{MSE}^2 = \frac{2}{W}\sum_{i=1}^{W/2} SE_{M+1-i}^2 \quad \overline{MSE}^3 = \frac{2}{W}\sum_{i=1}^{W/2} SE_{M+1-i}^3 \quad \overline{MSE}^4 = \frac{2}{W}\sum_{i=1}^{W/2} SE_{M+1-i}^4$$

$$\overline{MSE}^5 = \frac{2}{W}\sum_{i=1}^{W/2} SE_{M+1-i}^5 \qquad (62)$$

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:

$$\begin{gathered} \widehat{V}_{A+I}^x = P^x(V_S, V_A) \\ x = Index\left(\min\left[\overline{MSE}^x\right]\right) = Index\left(\min\left[\overline{MSE}^1, \overline{MSE}^2, \overline{MSE}^3, \overline{MSE}^4, \overline{MSE}^5\right]\right) \end{gathered} \qquad (63)$$

Algorithm A3 is formulated as follows:

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 $\widehat{V}_M^x, \widehat{V}_{M+1}^x, \widehat{V}_{M+2}^x, \ldots \widehat{V}_{M+i}^x \ldots \widehat{V}_A^x$, determine the number of times N^x for which the squared Euclidean distance $SE_{M+i}^x = \left(V_{M+i} - \widehat{V}_{M+i}^x\right)^2$ 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 to 5 respectively.

3. Predict the value at time T_{A+I} with the predictor, P^x, having the highest count Nx for the predicted values between time T_M and T_A:

$$\begin{gathered} \widehat{V}_{A+I}^x = P^x(V_S, V_A) \\ x = Index[\max(N1, N2, N3, N4, N5)] \end{gathered} \qquad (64)$$

2.5.3 Software Implementation of the Adaptive Weather Forecasting Algorithms

Figure 2.22 demonstrates a binary tree organisation of the Java program’s methods developed for the application of weather forecasting.

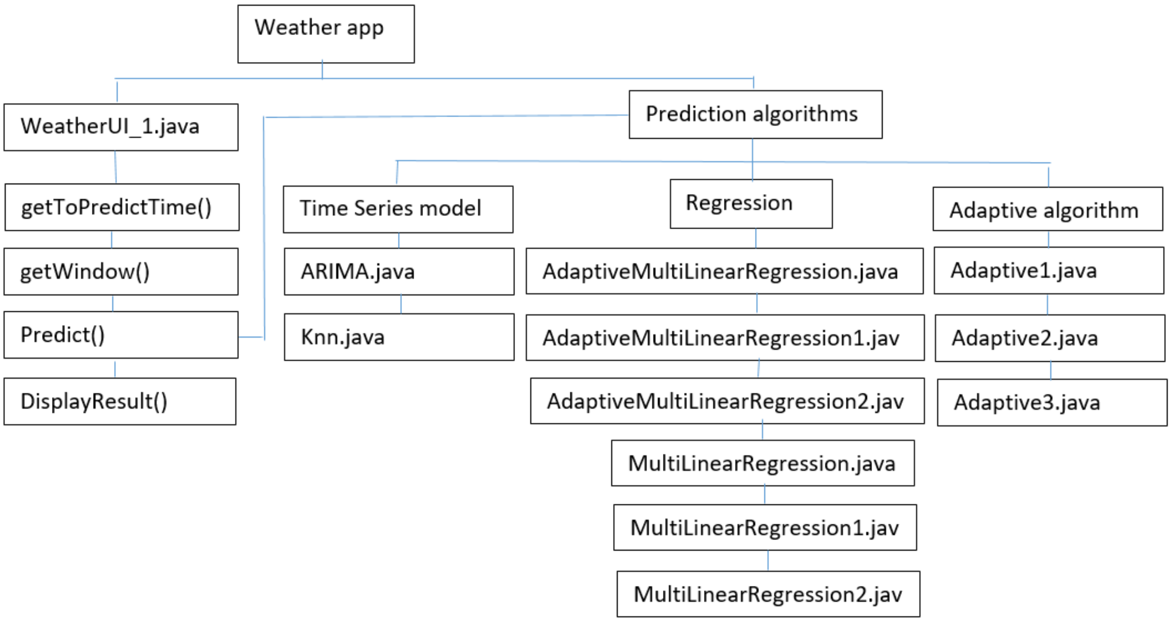


Figure 2.21: Program method structure

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 Figure 2.22. 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 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.

Figure 2.22 shows the proposed graphical user interface for the weather forecasting application. The user interface shows the actual weather data (temperature, humidity, luminosity, rainfall, pressure, wind speed and wind direction) in real time. The user data can set the time and window size for predicting weather. This user interface will then display all predicted values calculated by using the following prediction algorithms: ARIMA, KNN , adaptive multiple linear regression, adaptive multiple linear regression 1, adaptive multiple linear regression 2, multiple linear regression, multiple linear regression 1, multiple linear regression 2, adaptive 1, adaptive 2 and adaptive 3.

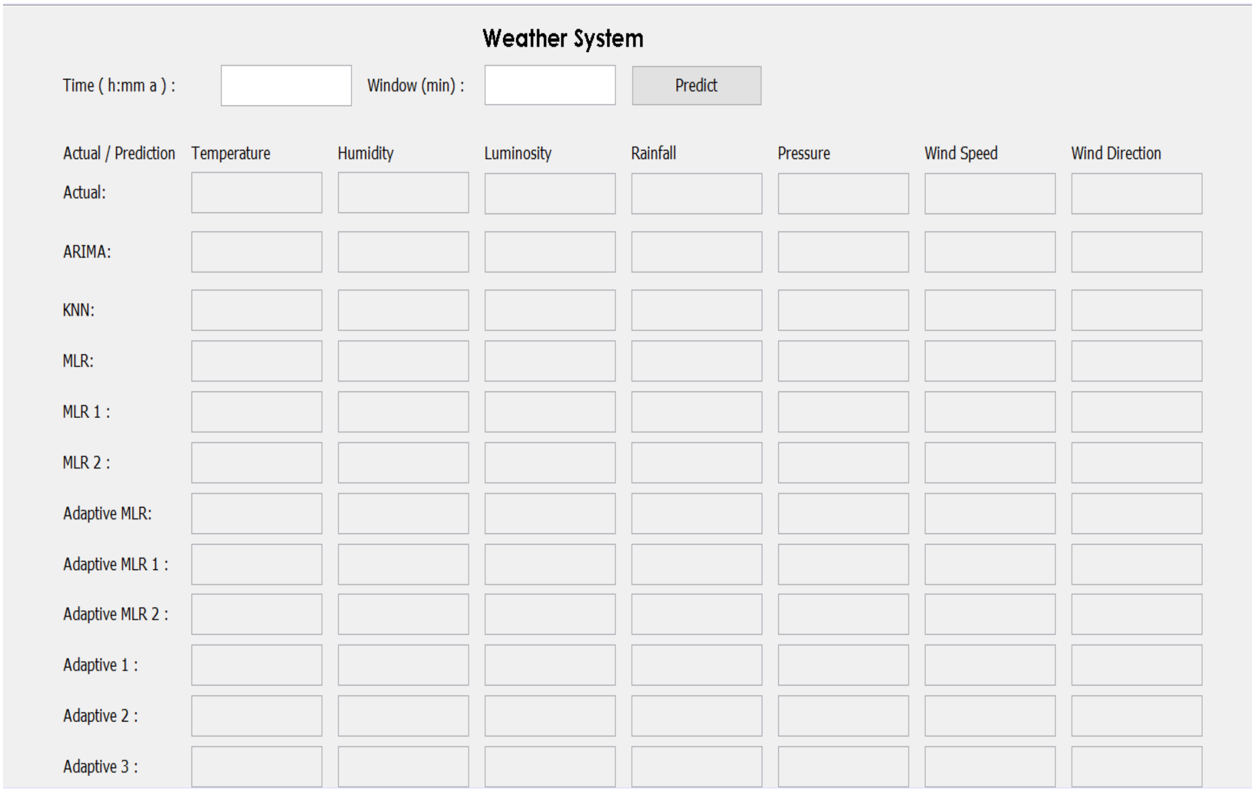


Figure 2.22: Graphical User Interface

CHAPTER 3

IMPLEMENTATION OF THE FORECASTING SYSTEM

In this work we have implemented three different forecasting systems. Two of them are microcontroller based and the third one is based on an integrated IoT weather forecasting device from Davis Instruments. The forecasting device from Davis Instruments comes into two versions namely cabled and wireless. In this work the cabled version has been used because the frequency range of the wireless version falls within a band restricted by the ICTA. Moreover, the frequency ranges of all other wireless IoT devices for weather forecasting were outside the bands allowed by the ICTA. Otherwise, a wireless integrated IoT device could have fitted this system very effectively.

3.1 Micro-controller based weather forecasting system with wireless gateway

The block diagram of the microprocessor based forecasting system with wireless gateway is shown in Figure 3.1.

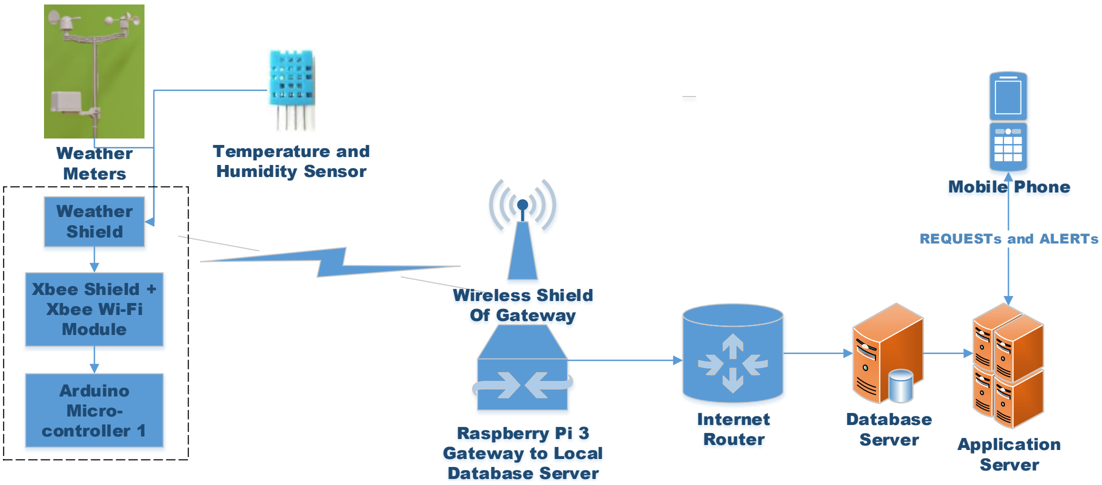

Weather Meters
Temperature and Humidity Sensor
Weather Shield
Xbee Shield + Xbee Wi-Fi Module
Arduino Micro-controller 1
Wireless Shield Of Gateway
Raspberry Pi 3 Gateway to Local Database Server
Internet Router
Database Server
Application Server
Mobile Phone
REQUESTs and ALERTs

Figure 3.1: Block diagram of the proposed weather forecasting system

This 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 two weather-monitoring nodes and acts as a gateway to the local database server. 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.

3.1.1 Arduino UNO Circuit Set-up and Programming

An Arduino board consists of a physical microcontroller, which contains a Central Processing Unit (CPU) along with a memory and programmable input and output peripherals. The Arduino IDE is used to write and upload the program code to the board via a USB cable. The Arduino program is commonly known as a sketch which is written in C/C++ programming language and has essentially two main functions namely the Setup and Loop. The Setup function runs only once and performs configuration tasks like setting pin modes and initializing imported libraries. The loop function runs consecutively controlling the Arduino board, and the external shields mounted [59].

The Arduino UNO board is used in the development of this project since it is the most traditional board and has been tested for a wide range of applications. The Arduino UNO board consists of the ATmega328 microcontroller chipset [60] as shown in Figure 3.2.

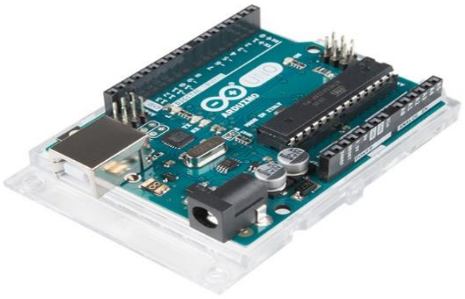


Figure 22: Arduino Board [60]

3.1.2 Raspberry-Pi Circuit Set-up and Programming

The Raspberry-Pi 3 is a powerful microcontroller with built-in Wi-Fi modules to allow the collection of data transmitted from the weather monitoring node. The Raspberry-Pi is intended for high compute needs and act as gateway by consolidating the collected data before transmission to the Database Server. The Raspberry Pi 3 is the third generation Raspberry Pi which has a 1.2GHz 64-bit quad-core ARMv8 CPU, 802.11n Wireless LAN, Bluetooth 4.1, Bluetooth Low Energy (BLE), 1GB RAM, 4 USB ports, 40 GPIO pins, Full HDMI port, Ethernet port, Combined 3.5mm audio jack and composite video, Camera interface (CSI), Display interface (DSI), Micro SD card slot (now push-pull rather than push-push), and VideoCore IV 3D graphics core [61].

Figure 3.3 shows the Raspberry-Pi 3. A power supply is needed to operate the Raspberry-Pi. Additionally, a MicroSD card is required to act as the hard drive of the Raspberry-Pi for holding the Operating System and the applications created. Typically, 8 GB or more is recommended for the MicroSD card. The OS, which is used on the Raspberry-Pi, is the Raspbian Jessie OS, which is a linux distribution and comes with Node-Red [62].

In order to transfer the OS to the MicroSD card, a MicroSD adapter or a MicroSD USB adapter is required. The initial set-up of the Raspberry Pi requires a monitor with HDMI input, HDMI cable, and USB keyboard and mouse. During the OS installation, I2C, which is a multi-device pass used to connect other devices such as SenseHAT or the Arduino, needs to be enabled.

In order to prevent latencies in the transmission of the collected data to the database server, an Ethernet cable is used to connect to an Internet router instead of Wi-Fi.

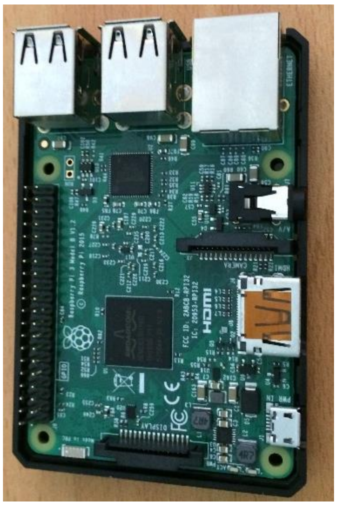


Figure 3.3: Raspberry Pi 3 Microcontroller

In this project, the Raspberry Pi 3 Microcontroller has been configured as a wireless router where the sensor node sends its data through the wireless interface. Upon intercepting the sensor values, the Raspberry Pi 3 forwards them through the Ethernet interface to be saved in the database.

The values recorded by the sensors of the weather monitoring nodes are temperature, humidity, atmospheric pressure, luminosity, wind speed, wind direction, and rainfall. Figure 3.4 illustrates how the sensor values are recorded and transmitted. The sensor values obtained from the nodes are stored in JSON format and transmitted through the wireless link to the Raspberry Pi for aggregation.

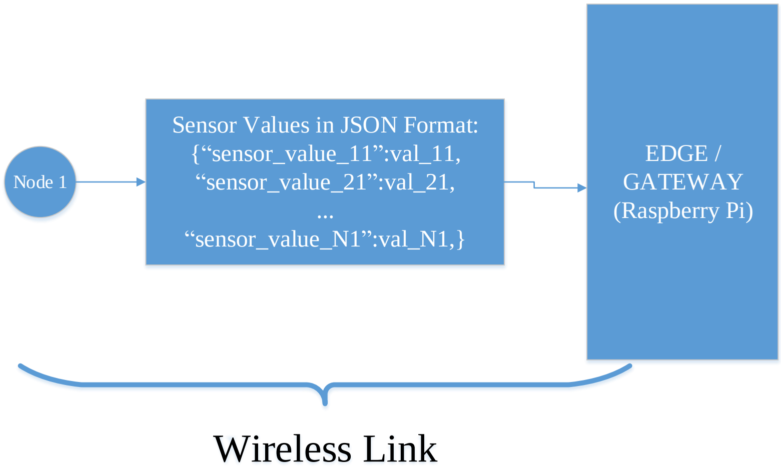

Node 1
Sensor Values in JSON Format:
{“sensor_value_11”:val_11,
“sensor_value_21”:val_21,
...
“sensor_value_N1”:val_N1,}
EDGE / GATEWAY (Raspberry Pi)
Wireless Link

Figure 3.4: Sensor values recorded

3.1.3 X-Bee Shield Set-up

XBees are tiny blue chips that can communicate wirelessly with each other. XBee modules provide flexibility and are obtainable in both surface mount and through-hole formats. The XBee Wi-Fi shares a common imprint with other XBee modules therefore enabling different XBee technologies to be drop-in substitutes [63].

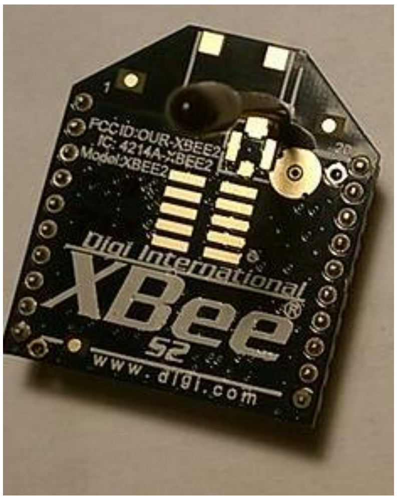


Figure 3.5: XBee module

The XBee Shield, shown in Figure 3.6, connects straight to any development board that has an Arduino imprint and outfits it with wireless communication abilities using the XBee module [63].

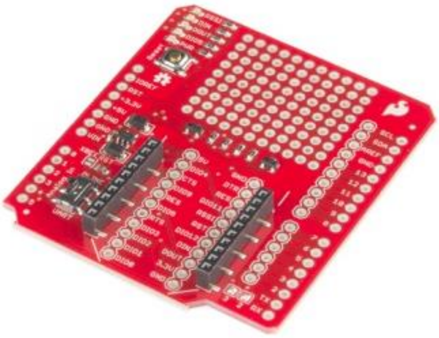


Figure 3.6: XBee Shield

Figure 3.7 shows how the XBee shield has been used in the project. The XBee shield is mounted on the Arduino Uno. The weather shield is mounted onto the XBee shield. The Xbee Shield allows data from the weather shield and meters to be transmitted to the Raspberry Pi 3 using Wi-Fi.

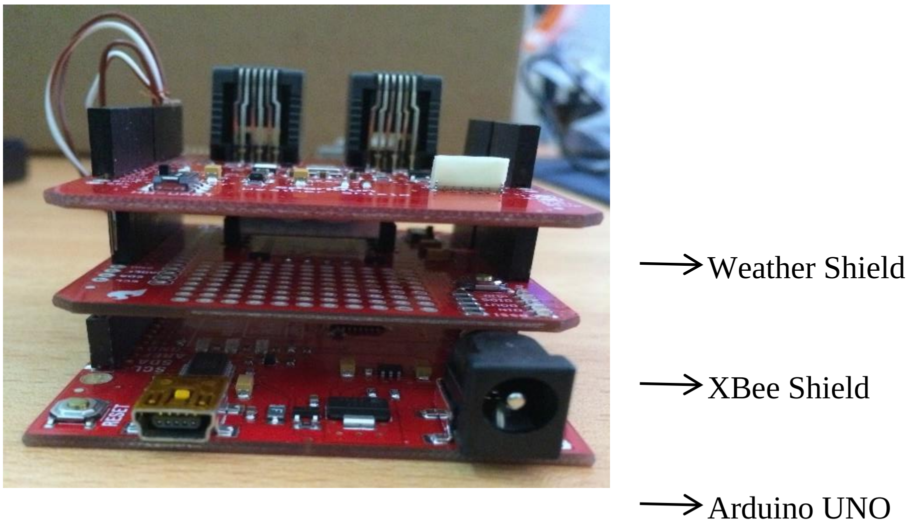


Figure 3.7: XBee Shield setup with Arduino Uno and Weather Shield

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 Figure 3.8.

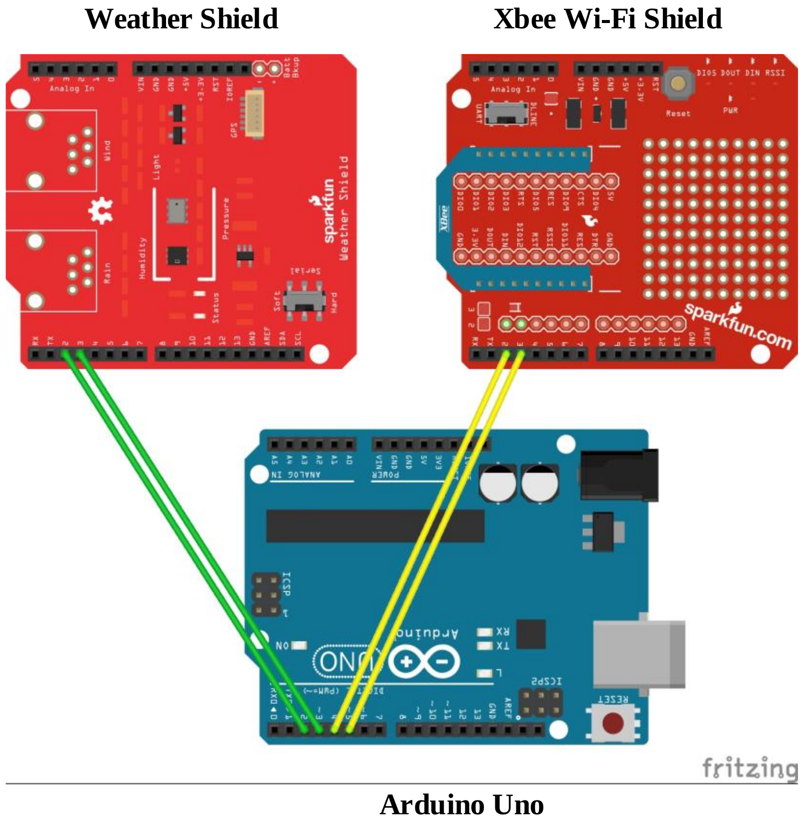


Figure 3.8: Re-wiring between XBee Shield, Arduino Uno and Weather Shield

3.1.4 Weather Sensors

A set of sensors is used to capture weather information. The SparkFun Atmospheric Sensor Breakout and the SparkFun Weather Meters are used in this project.

3.1.4.1 Atmospheric Sensors: Temperature, Humidity and Barometric Pressure

The SparkFun Atmospheric Sensor Breakout has been used in this project. It features a humidity/temperature sensor, barometric pressure sensor and a light sensor. The shield is connected to the Arduino Uno. The sensor has a temperature range of -45C to 80C, a humidity range of 0-100% and a pressure range of 30,000Pa to 110,000Pa with a relative accuracy of 12Pa [64].

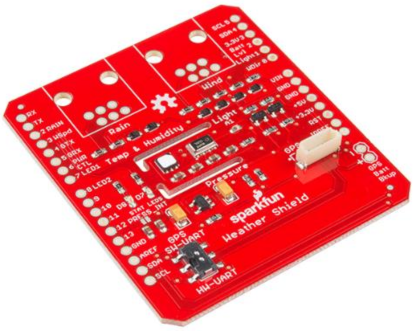


Figure 3.9: SparkFun Atmospheric Sensor Breakout

3.1.4.2. Weather meters: Wind Vane, Cup Anemometer and Rain Gauge

The SparkFun Weather Meters measure wind direction, wind speed and rainfall. The rain gauge is a self-emptying bucket-type rain gauge, which leads to a trigger for every 0.011-inch of rain that is accumulated. The anemometer translates the wind speed by simply shutting a key which each revolution. A wind speed of 1.492 MPH creates one closure per second. The wind vane records wind direction as a voltage. The voltage is created by the arrangement of resistors inside the sensor. The vane’s magnet may shut two switches at once, allowing up to 16 distinctive positions to be specified. The weather meters are connected to the SparkFun weather shield [65].

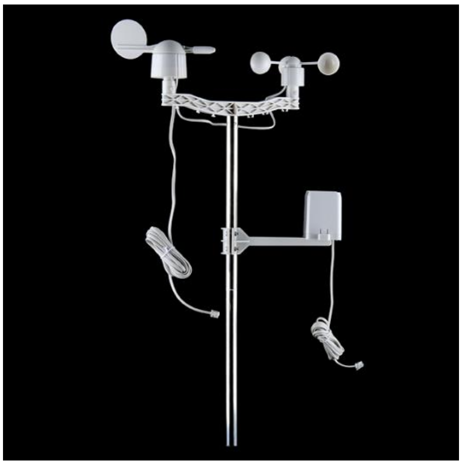
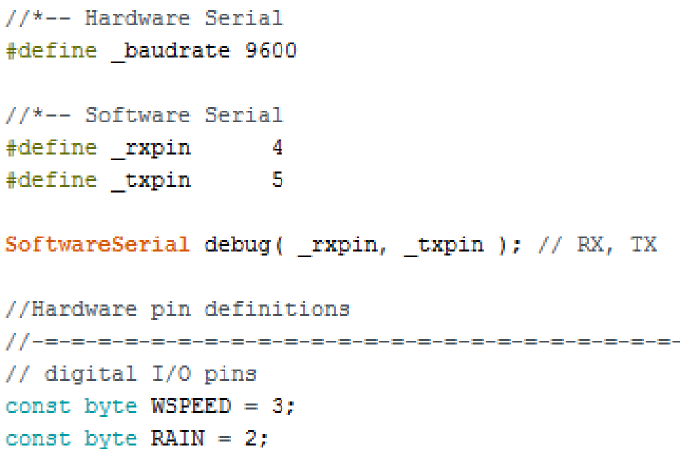


Figure 3.10: SparkFun Weather Meters

3.1.5 Software Design for Arduino Codes

As mentioned in the previous sections, the overlapping of the pins 2 and 3 for the XBee Wi-Fi shield and Weather shield has been overcome at the hardware level by performing a re-wiring.

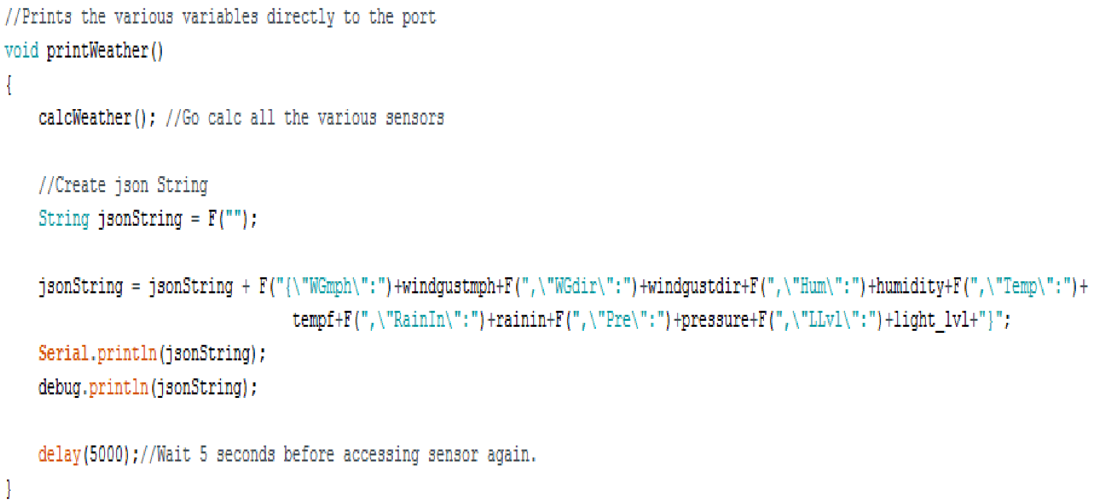
The XBee Wi-Fi shield has been made to use the pins 4 and 5 to allow the Weather shield to use the default pins 2 and 3. The Arduino code for this manipulation is as shown in Figure 3.11.



```
//*-- Hardware Serial
#define _baudrate 9600

//*-- Software Serial
#define _rxpin      4
#define _txpin      5

SoftwareSerial debug( _rxpin, _txpin ); // RX, TX

//Hardware pin definitions
//-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
// digital I/O pins
const byte WSPEED = 3;
const byte RAIN = 2;
```

Figure 3.11: Arduino code representing the re-wiring performed

Once the wireless communication between the XBee Wi-Fi module and raspberry pi is set-up, sensor values are transmitted over the air-interface to be aggregated at the edge before being stored in the database. The Arduino code for writing the sensor values into a JSON string and transmit through the XBee Wi-Fi module is as depicted in Figure 3.12.

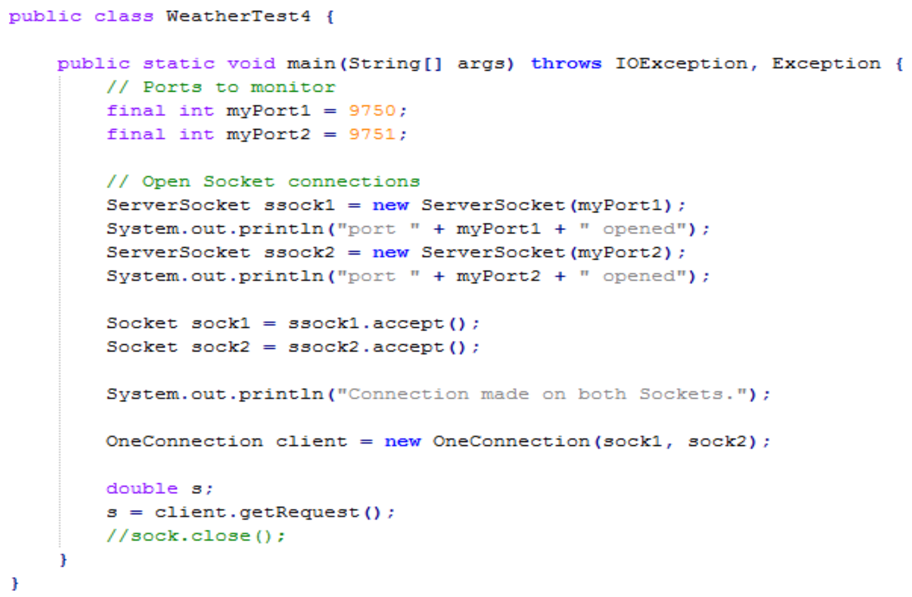


```
//Prints the various variables directly to the port
void printWeather()
{
    calcWeather(); //Go calc all the various sensors

    //Create json String
    String jsonString = F("");

    jsonString = jsonString + F("{\"WGmph\":")+windgustmph+F(",\"WGdir\":")+windgustdir+F(",\"Hum\":")+humidity+F(",\"Temp\":")+
                                tempf+F(",\"RainIn\":")+rainin+F(",\"Pre\":")+pressure+F(",\"LLvl\":")+light_lvl+"}";
    Serial.println(jsonString);
    debug.println(jsonString);

    delay(5000);//Wait 5 seconds before accessing sensor again.
}
```

Figure 3.12: Arduino code for transmission of sensor data in a JSON string

3.1.6 Software Design for Java Codes running on Raspberry Pi

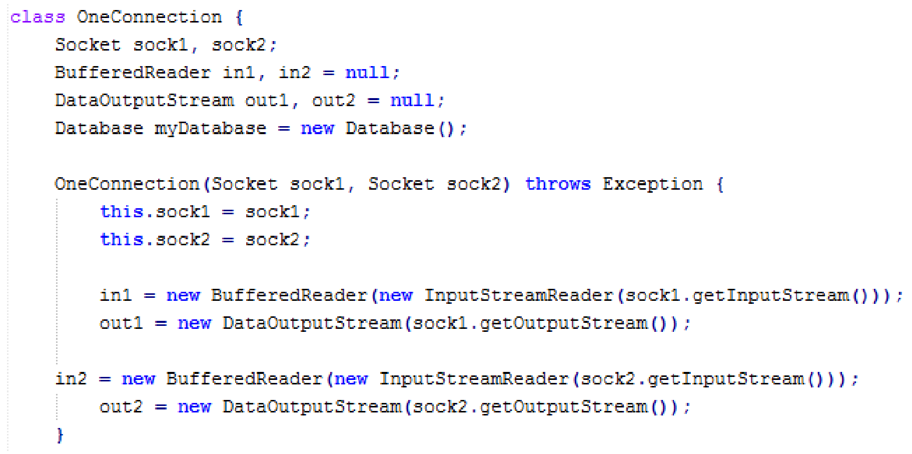
The program for intercepting, aggregating and inserting into database the sensor values; has been written in JAVA programming language. The main class is written as shown in Figure 3.13.



```
public class WeatherTest4 {

    public static void main(String[] args) throws IOException, Exception {
        // Ports to monitor
        final int myPort1 = 9750;
        final int myPort2 = 9751;

        // Open Socket connections
        ServerSocket ssock1 = new ServerSocket(myPort1);
        System.out.println("port " + myPort1 + " opened");
        ServerSocket ssock2 = new ServerSocket(myPort2);
        System.out.println("port " + myPort2 + " opened");

        Socket sock1 = ssock1.accept();
        Socket sock2 = ssock2.accept();

        System.out.println("Connection made on both Sockets.");

        OneConnection client = new OneConnection(sock1, sock2);

        double s;
        s = client.getRequest();
        //sock.close();
    }
}
```

Figure 3.13: Main Class

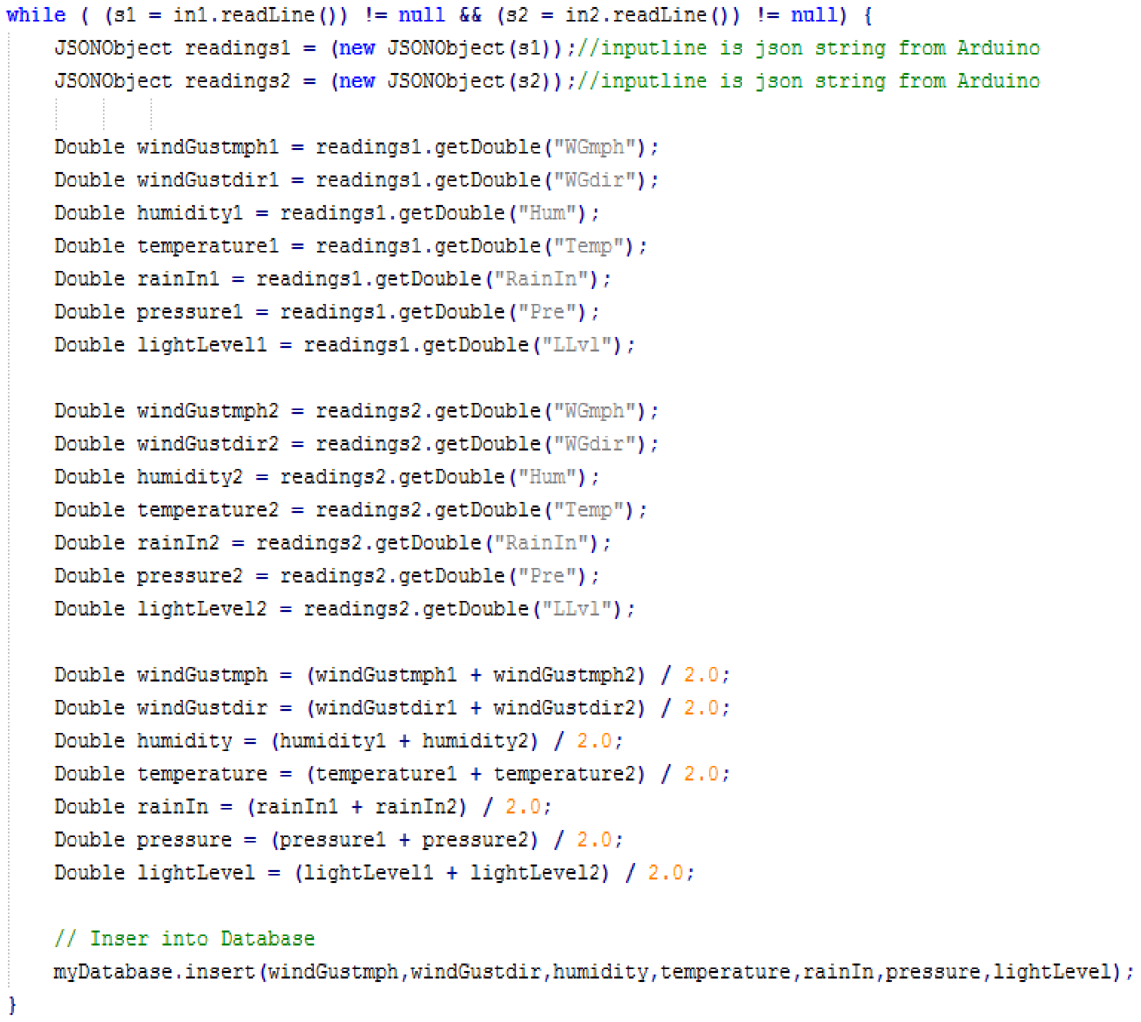
The main class consists of the codes to listen for wireless communications on the TCP ports 9750, and 9751 where each TCP port is configured at each sensor node. A socket listener for each node waits for communication to be established before calling the “OneConnection” function which processes the data streams as shown in Figure 3.14.



```
class OneConnection {
    Socket sock1, sock2;
    BufferedReader in1, in2 = null;
    DataOutputStream out1, out2 = null;
    Database myDatabase = new Database();

    OneConnection(Socket sock1, Socket sock2) throws Exception {
        this.sock1 = sock1;
        this.sock2 = sock2;

        in1 = new BufferedReader(new InputStreamReader(sock1.getInputStream()));
        out1 = new DataOutputStream(sock1.getOutputStream());

    in2 = new BufferedReader(new InputStreamReader(sock2.getInputStream()));
        out2 = new DataOutputStream(sock2.getOutputStream());
    }
```

Figure 3.14: Connection Class

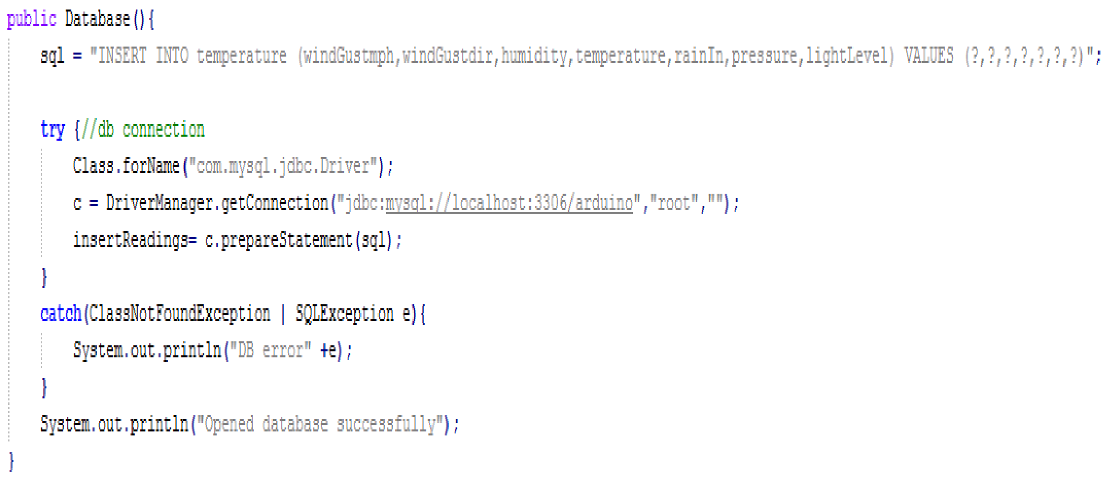
The function “OneConneciton” launches the connection to the database and establishes the input and output buffered streams on both socket connections. The data being transmitted on these streams are then read and processed as shown in Figure 3.15.



```
while ( (s1 = in1.readLine()) != null && (s2 = in2.readLine()) != null) {
    JSONObject readings1 = (new JSONObject(s1));//inputline is json string from Arduino
    JSONObject readings2 = (new JSONObject(s2));//inputline is json string from Arduino

    Double windGustmph1 = readings1.getDouble("WGmph");
    Double windGustdir1 = readings1.getDouble("WGdir");
    Double humidity1 = readings1.getDouble("Hum");
    Double temperature1 = readings1.getDouble("Temp");
    Double rainIn1 = readings1.getDouble("RainIn");
    Double pressure1 = readings1.getDouble("Pre");
    Double lightLevel1 = readings1.getDouble("LLvl");

    Double windGustmph2 = readings2.getDouble("WGmph");
    Double windGustdir2 = readings2.getDouble("WGdir");
    Double humidity2 = readings2.getDouble("Hum");
    Double temperature2 = readings2.getDouble("Temp");
    Double rainIn2 = readings2.getDouble("RainIn");
    Double pressure2 = readings2.getDouble("Pre");
    Double lightLevel2 = readings2.getDouble("LLvl");

    Double windGustmph = (windGustmph1 + windGustmph2) / 2.0;
    Double windGustdir = (windGustdir1 + windGustdir2) / 2.0;
    Double humidity = (humidity1 + humidity2) / 2.0;
    Double temperature = (temperature1 + temperature2) / 2.0;
    Double rainIn = (rainIn1 + rainIn2) / 2.0;
    Double pressure = (pressure1 + pressure2) / 2.0;
    Double lightLevel = (lightLevel1 + lightLevel2) / 2.0;

    // Inser into Database
    myDatabase.insert(windGustmph,windGustdir,humidity,temperature,rainIn,pressure,lightLevel);
}
```

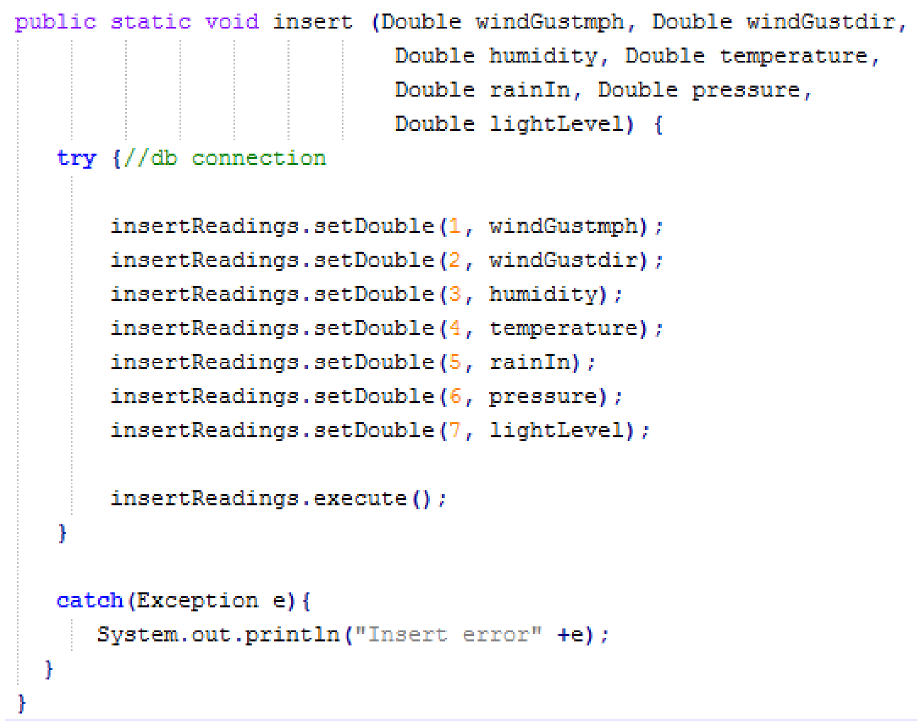
Figure 3.15: Code for aggregating sensor values and inserting into the database

The sensor values intercepted in JSON format are stripped after being read from the different data streams flowing through the socket connections. After obtaining the individual sensor values from both sensor nodes, an averaging is performed and then the “insert” function is called to save these values into the database. Figure 3.16 shows the code used for the setup of the connection to the database. Figure 3.17 shows the code snippet for the section where the aggregated sensor values are inserted into the database.



```java
public Database(){
    sql = "INSERT INTO temperature (windGustmph,windGustdir,humidity,temperature,rainIn,pressure,lightLevel) VALUES (?,?,?,?,?,?,?)";

    try {//db connection
        Class.forName("com.mysql.jdbc.Driver");
        c = DriverManager.getConnection("jdbc:mysql://localhost:3306/arduino","root","");
        insertReadings= c.prepareStatement(sql);
    }
    catch(ClassNotFoundException | SQLException e){
        System.out.println("DB error" +e);
    }
    System.out.println("Opened database successfully");
}
```

Figure 3.16: Code for establishment of connection to database



```java
public static void insert (Double windGustmph, Double windGustdir,
                           Double humidity, Double temperature,
                           Double rainIn, Double pressure,
                           Double lightLevel) {
   try {//db connection

       insertReadings.setDouble(1, windGustmph);
       insertReadings.setDouble(2, windGustdir);
       insertReadings.setDouble(3, humidity);
       insertReadings.setDouble(4, temperature);
       insertReadings.setDouble(5, rainIn);
       insertReadings.setDouble(6, pressure);
       insertReadings.setDouble(7, lightLevel);

       insertReadings.execute();
   }

   catch(Exception e){
      System.out.println("Insert error" +e);
   }
}
```

Figure 3.17: Code for inserting the aggregated sensor values into database

3.1.7 Web Server Architecture and Database System

The structure of the proposed web server is given in Figure 3.18.

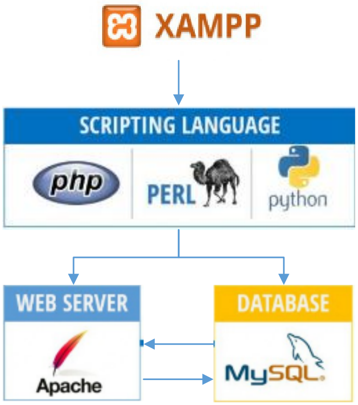


Figure 3.18: Architecture of the Web Server

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 Figure 3.19.

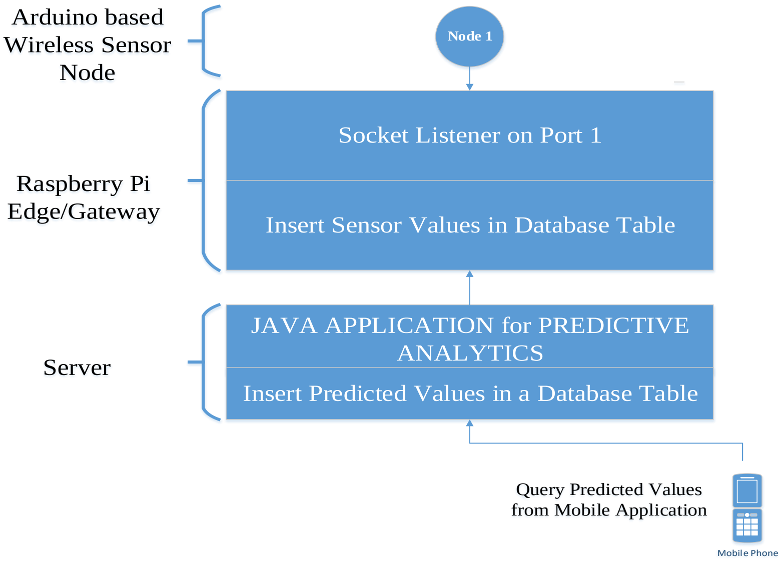
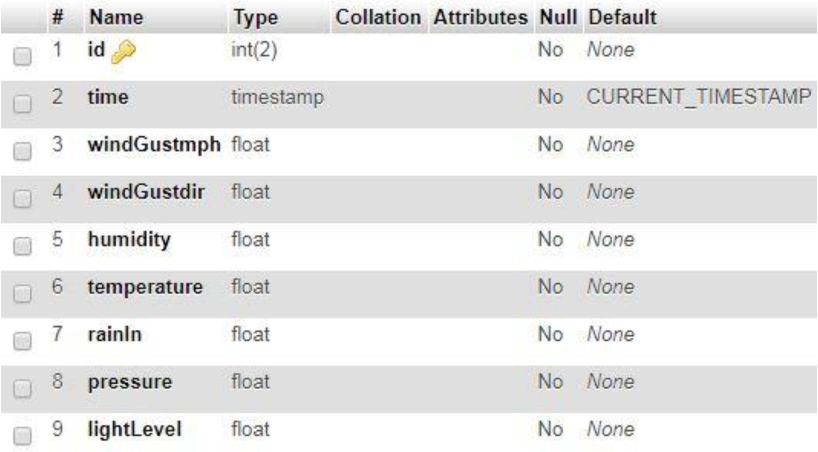

Arduino based Wireless Sensor Node
Node 1
Raspberry Pi Edge/Gateway
Socket Listener on Port 1
Insert Sensor Values in Database Table
Server
JAVA APPLICATION for PREDICTIVE ANALYTICS
Insert Predicted Values in a Database Table
Query Predicted Values from Mobile Application
Mobile Phone

Figure 3.19: Architecture for data acquisition and query from mobile device

The Arduino based wireless sensor node transmits the collected weather data to the Raspberry Pi 3. The latter aggregates the data which are then inserted in the database table. The server uses a Java application for predictive analytics to calculate forecasted weather data. The data is then stored in another table of the same database. A user can then query the predicted values using an application running on a mobile phone.

3.1.8 Database Architecture

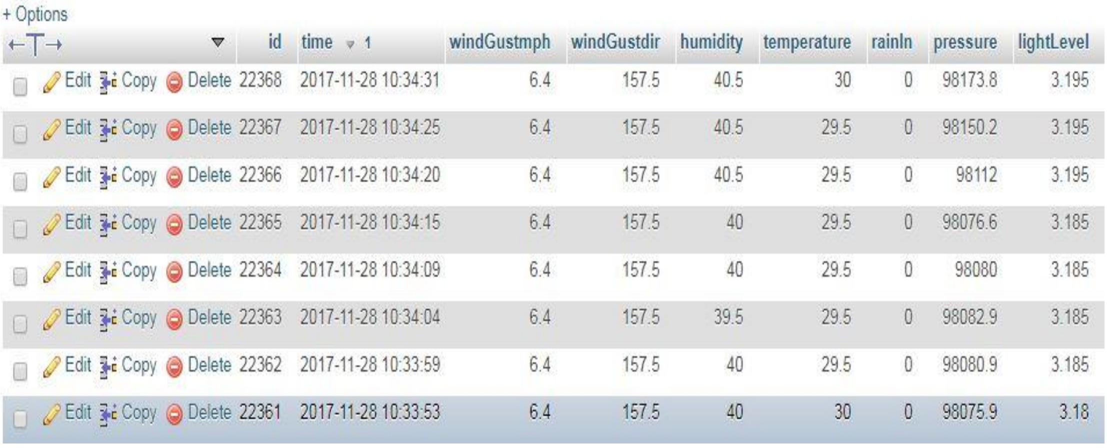
Two tables are implemented in a single database for the proposed system. One table is used for storing the aggregated sensor values and the second one is used to save predicted values which are accessed by the mobile application. The structure of the Sensor values record table are as shown in Figure 3.20.



	#	Name	Type	Collation	Attributes	Null	Default
☐	1	**id** 🔑	int(2)			No	*None*
☐	2	**time**	timestamp			No	CURRENT_TIMESTAMP
☐	3	**windGustmph**	float			No	*None*
☐	4	**windGustdir**	float			No	*None*
☐	5	**humidity**	float			No	*None*
☐	6	**temperature**	float			No	*None*
☐	7	**rainIn**	float			No	*None*
☐	8	**pressure**	float			No	*None*
☐	9	**lightLevel**	float			No	*None*

Figure 3.20: Sensor values record table Structure

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 Figure 3.21.



☐		id	time ▾ 1	windGustmph	windGustdir	humidity	temperature	rainIn	pressure	lightLevel
☐	Edit Copy Delete	22368	2017-11-28 10:34:31	6.4	157.5	40.5	30	0	98173.8	3.195
☐	Edit Copy Delete	22367	2017-11-28 10:34:25	6.4	157.5	40.5	29.5	0	98150.2	3.195
☐	Edit Copy Delete	22366	2017-11-28 10:34:20	6.4	157.5	40.5	29.5	0	98112	3.195
☐	Edit Copy Delete	22365	2017-11-28 10:34:15	6.4	157.5	40	29.5	0	98076.6	3.185
☐	Edit Copy Delete	22364	2017-11-28 10:34:09	6.4	157.5	40	29.5	0	98080	3.185
☐	Edit Copy Delete	22363	2017-11-28 10:34:04	6.4	157.5	39.5	29.5	0	98082.9	3.185
☐	Edit Copy Delete	22362	2017-11-28 10:33:59	6.4	157.5	40	29.5	0	98080.9	3.185
☐	Edit Copy Delete	22361	2017-11-28 10:33:53	6.4	157.5	40	30	0	98075.9	3.18

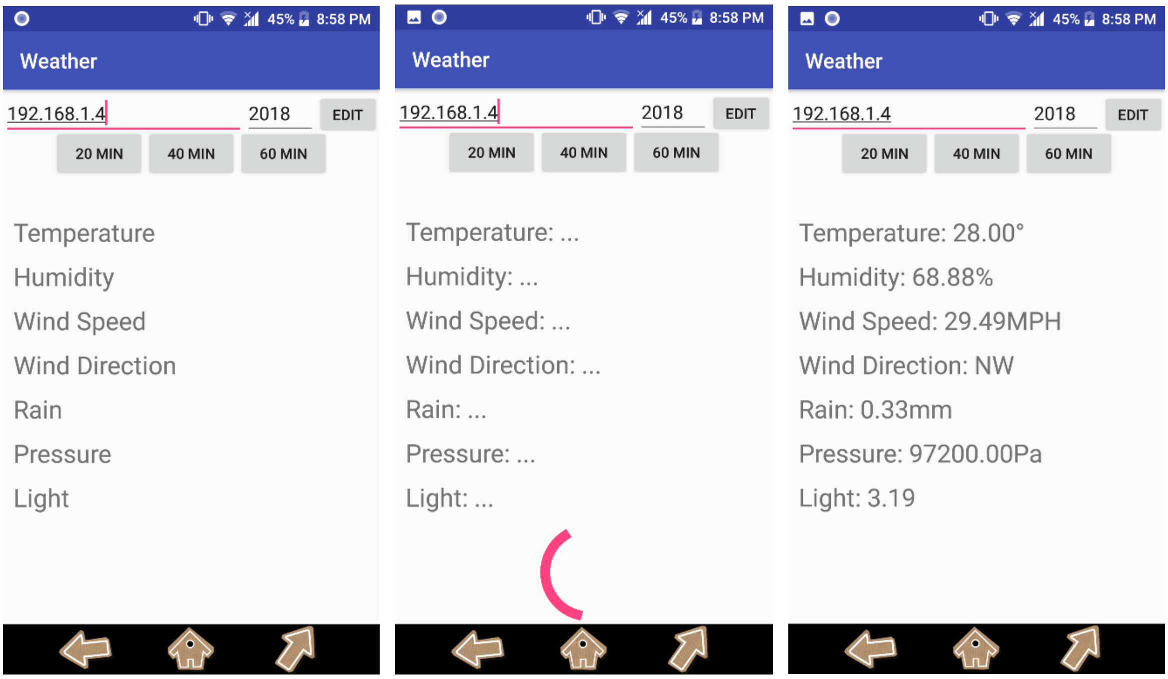
Figure 3.21: Sample of Recorded Sensor values

3.1.9 Control Station

The control station uses the data from the database server and runs the weather forecasting algorithms on them to give predictions. The predictions can be retrieved by the mobile application.

3.1.10 Mobile application development

The mobile application requests and acquires the predicted weather data from the application server through the socket connection established. The layout of the mobile application is shown in Figure 3.22.



192.168.1.4
2018
Temperature
Humidity
Wind Speed
Wind Direction
Rain
Pressure
Light

192.168.1.4
2018
Temperature: ...
Humidity: ...
Wind Speed: ...
Wind Direction: ...
Rain: ...
Pressure: ...
Light: ...

192.168.1.4
2018
Temperature: 28.00°
Humidity: 68.88%
Wind Speed: 29.49MPH
Wind Direction: NW
Rain: 0.33mm
Pressure: 97200.00Pa
Light: 3.19

Figure 3.22: Layout of mobile application

The flowchart of the mobile application is shown in Figure 3.23.

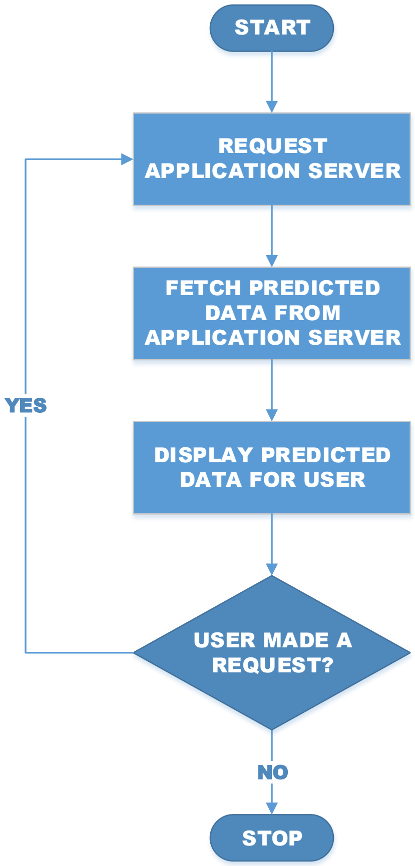

START
REQUEST APPLICATION SERVER
FETCH PREDICTED DATA FROM APPLICATION SERVER
DISPLAY PREDICTED DATA FOR USER
USER MADE A REQUEST?
YES
NO
STOP

Figure 3.23: Flowchart of mobile application

When user starts the mobile application, the application connects with the application server. Upon receiving specific prediction request from the mobile application, the application server connects to the IBM Cloudant database for retrieving relevant past data to use for the prediction. Prediction data is the pushed over to the mobile application to be displayed for the user.

3.2 Micro-controller based weather forecasting system with cabled connection

This setup can be adopted when the distance between the Arduino Microcontroller and the application server is relatively small (5-10m). In this case there no need for the Raspberry-pi and the X-bee shield. Moreover, it eliminates the need for a power-supply for the Arduino microcontroller which will be powered directly by the Application Server. This simplified set-up is shown in Figure 3.24:

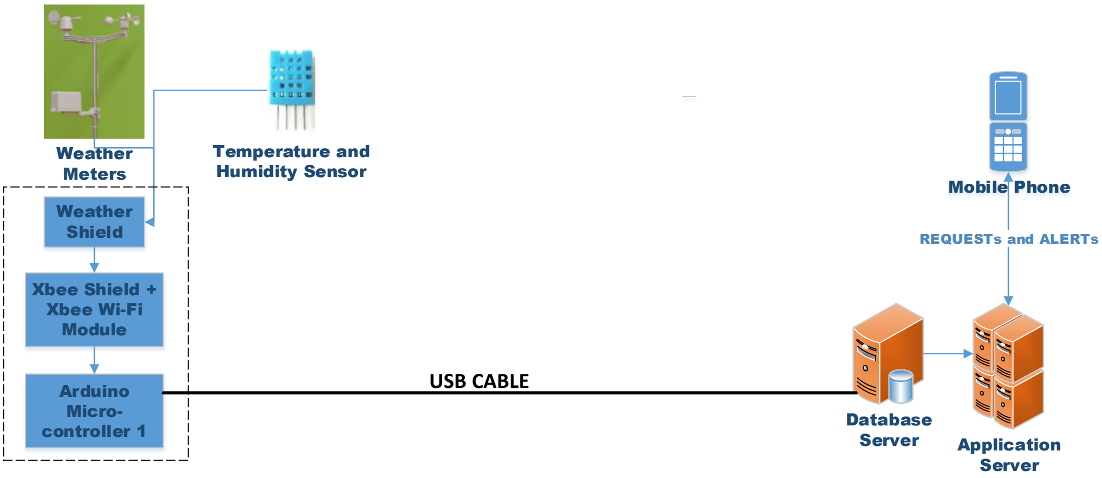


Figure 3.24: Weather forecasting system with cabled connection.

3.3 IoT based weather forecasting system with cabled connection

The microcontroller based weather forecasting systems are cheaper but their set-up is not user friendly and require expert knowledge. Moreover, the system is not very robust to strong winds and require significant amount of shielding. A more robust and convenient alternative can be found in the form of integrated weather sensors such as the WS-900-IP weather data monitoring station or the Cabled Davis Vantage-Pro2 weather station [66]. However, the WS-900-IP weather stations transmits at a frequency which is not allowed by the ICTA and in fact most wireless integrated weather monitoring devices that we have identified, transmit at frequencies that are not allowed by the ICTA. Consequently, the only choice left was the Cabled Davis Vantage-Pro2 weather station.

Figure 3.25 shows the setup with the Cabled Davis Vantage-Pro2 weather station.

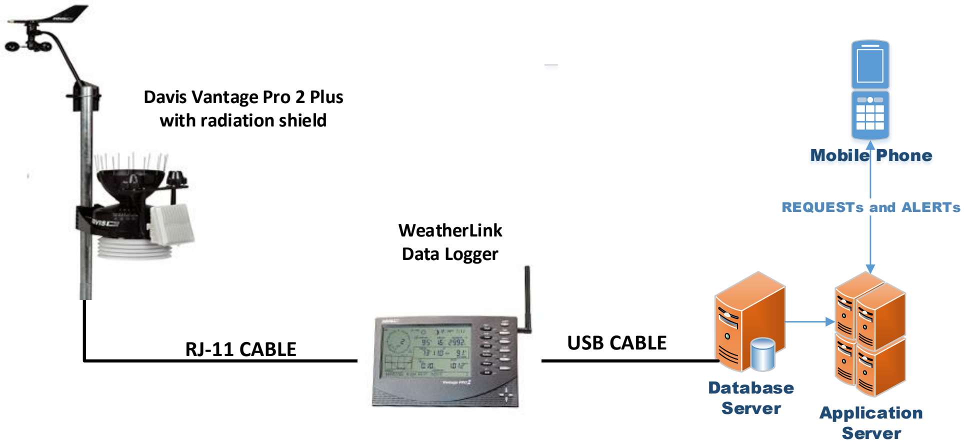


Figure 3.25 : Weather forecasting system with Davis Vantage Pro 2 plus [66].

The cabled Davis Vantage Pro 2 plus is equipped with sensors to measure temperature, humidity, atmospheric pressure, wind direction, wind speed, rainfall and luminosity. The Integrated Sensor Suite combines a rain collector, temperature and humidity sensors, anemometer, UV, and solar radiation sensors into one package. This is illustrated in Figure 3.26:

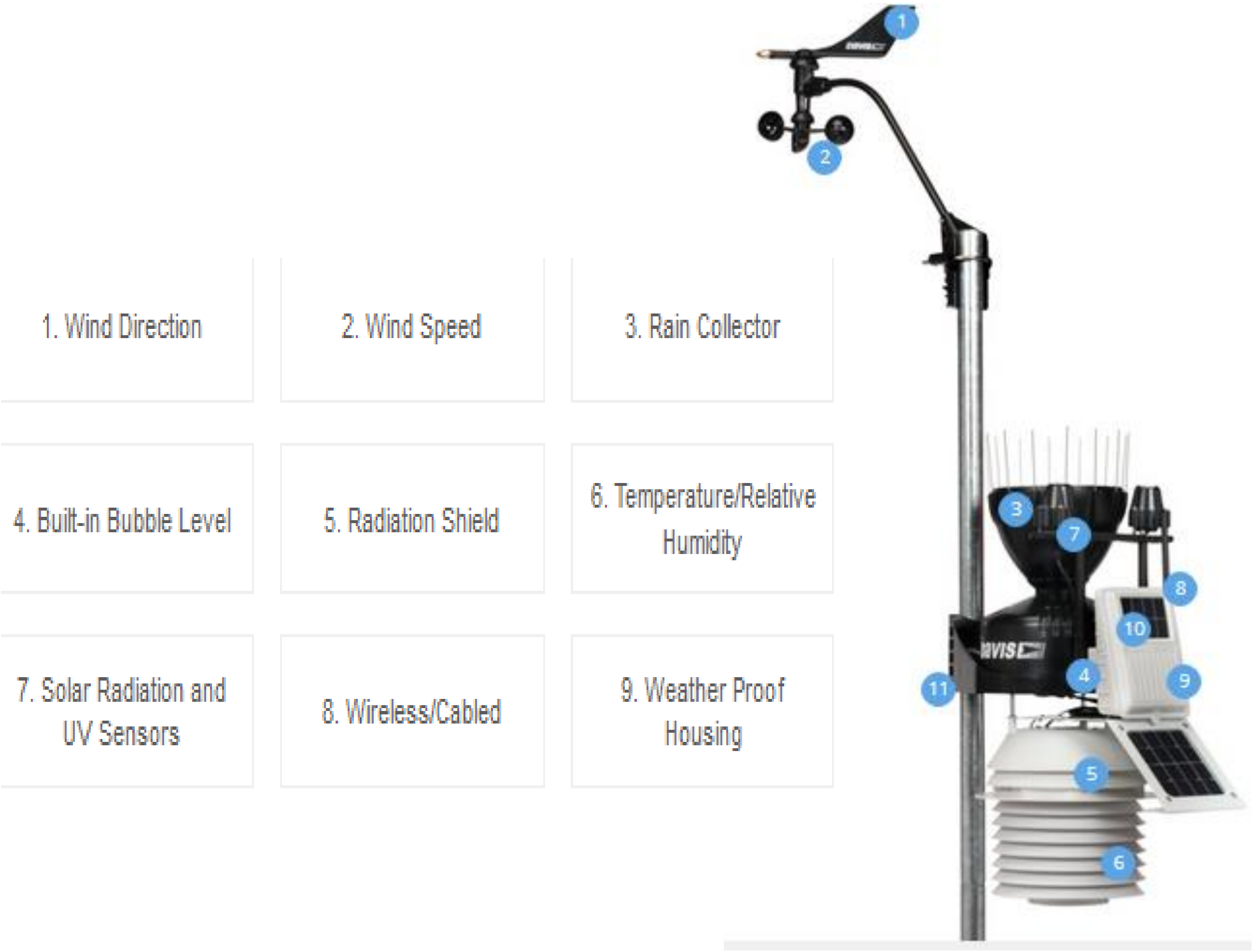


Figure 3.26: Davis Vantage Pro 2 plus sensors [66].

It can optionally include solar panel (the tenth label in the Figure 3.26). It also has a data logger / console which can be attached to the sensors to obtain the readings as shown in Figure 3.27. The readings are obtained at a fixed rate of one reading every 2.5 seconds. From the data logger the readings can be sent to the database server from where the processing is carried out in a similar way as the previous microcontroller based systems. The advantages of this station are that it is engineered to withstand scorching sun, corrosion, 200 mph (321 kmh) winds, temperature extremes, provide the highest level of accuracy, reliability and ruggedness.

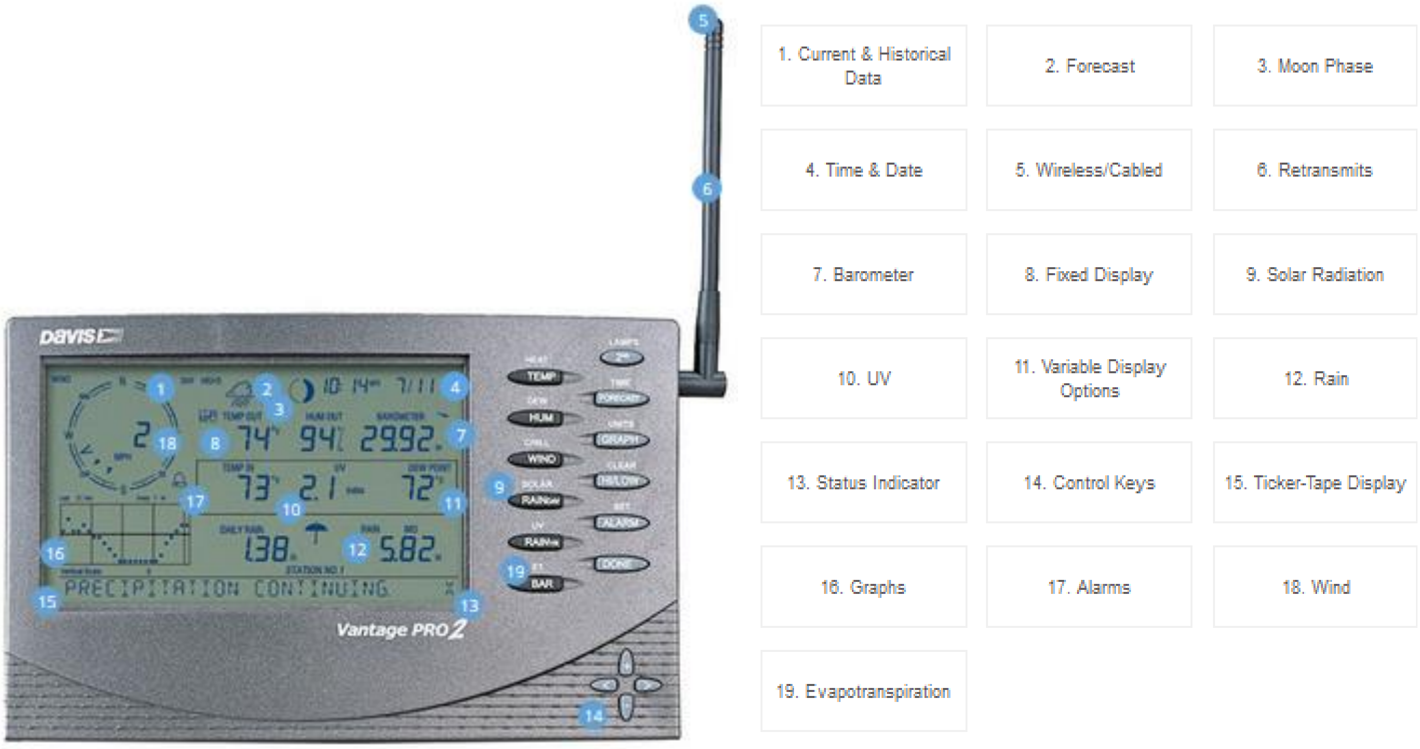


Figure 3.27: Vantage Pro 2 Console [66].

CHAPTER 4

CLOUD SERVER CONFIGURATION

The cloud-based system developed has been implemented in using two different types of sensorial nodes. These are:

1. IoT system using Arduino micro-controllers, weather meters, temperature and humidity sensor, and raspberry Pi.
2. System using the Davis Pro Vantage 2 plus system.

The cloud infrastructure used is: IBM Bluemix. The main service used is the Cloudant NoSQL database where massive real-time data is stored in different documents (like tables in SQL databases). The prediction algorithm coded in the JAVA programming language runs on the DevOps Insights service. The system is as shown in Figure 4.1.

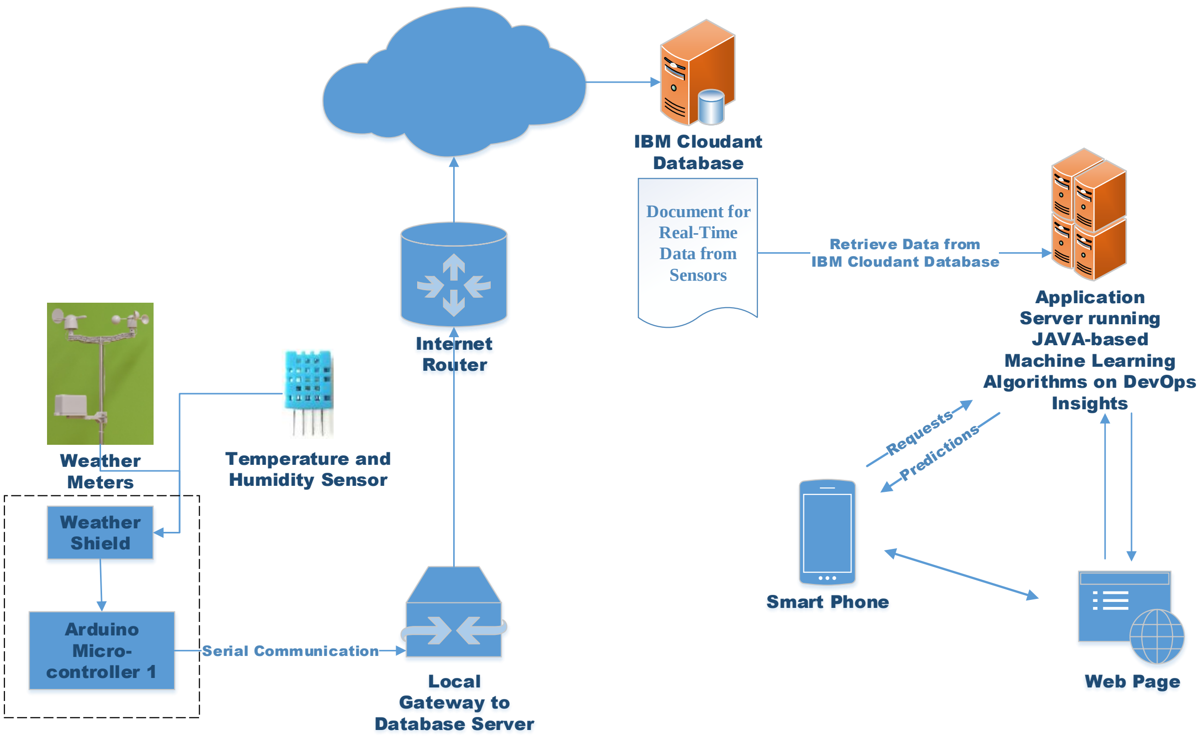

IBM Cloudant Database
Document for Real-Time Data from Sensors
Retrieve Data from IBM Cloudant Database
Application Server running JAVA-based Machine Learning Algorithms on DevOps Insights
Internet Router
Weather Meters
Temperature and Humidity Sensor
Weather Shield
Arduino Micro-controller 1
Serial Communication
Local Gateway to Database Server
Requests
Predictions
Smart Phone
Web Page

Figure 4.1: conventional IoT architecture

The system using the Davis Pro Vantage 2 plus system is as shown in Figure 4.2.

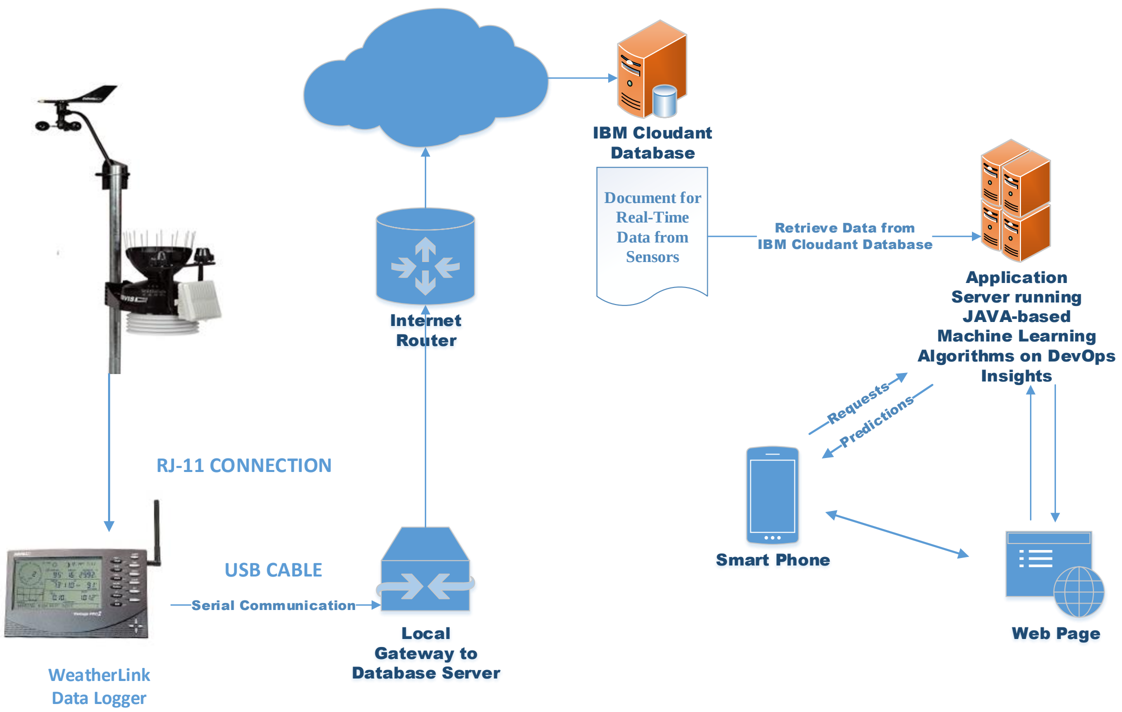
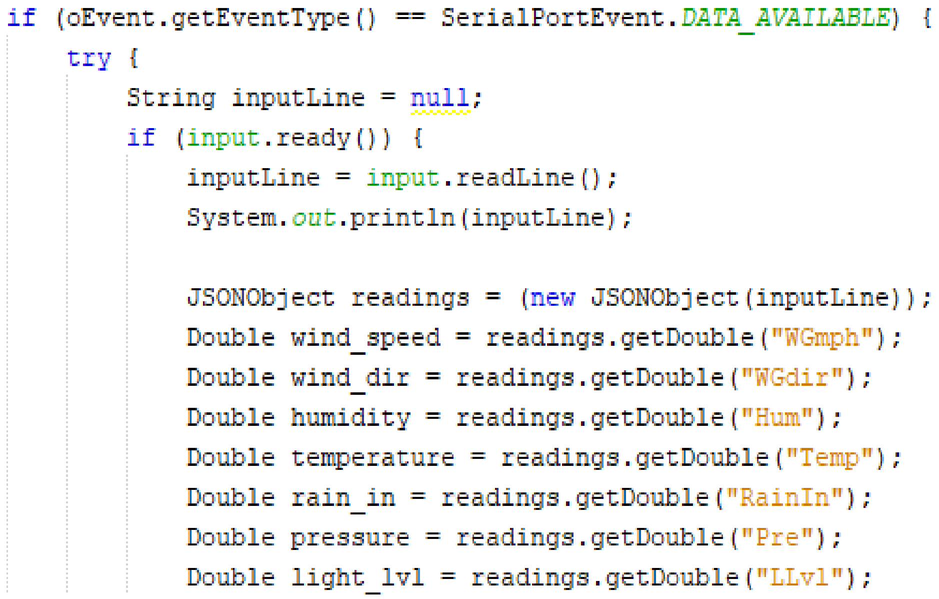

IBM Cloudant Database
Document for Real-Time Data from Sensors
Retrieve Data from IBM Cloudant Database
Application Server running JAVA-based Machine Learning Algorithms on DevOps Insights
Internet Router
Requests
Predictions
RJ-11 CONNECTION
Smart Phone
USB CABLE
Serial Communication
Web Page
Local Gateway to Database Server
WeatherLink Data Logger

Figure 4.2: Architecture with Davis Pro Vantage 2 Plus

The mobile device queries the application server through an established connection. Upon receiving the request, the application server connects to the IBM Cloudant database to acquire relevant past data to perform a prediction for the forthcoming 20, 40, or 60 minutes depending on the user selection on the mobile application. Once, the prediction is performed, the predicted data is sent over the same connection back to the requestor mobile application to be displayed for the user.

4.1 Data capture codes for Arduino-based system

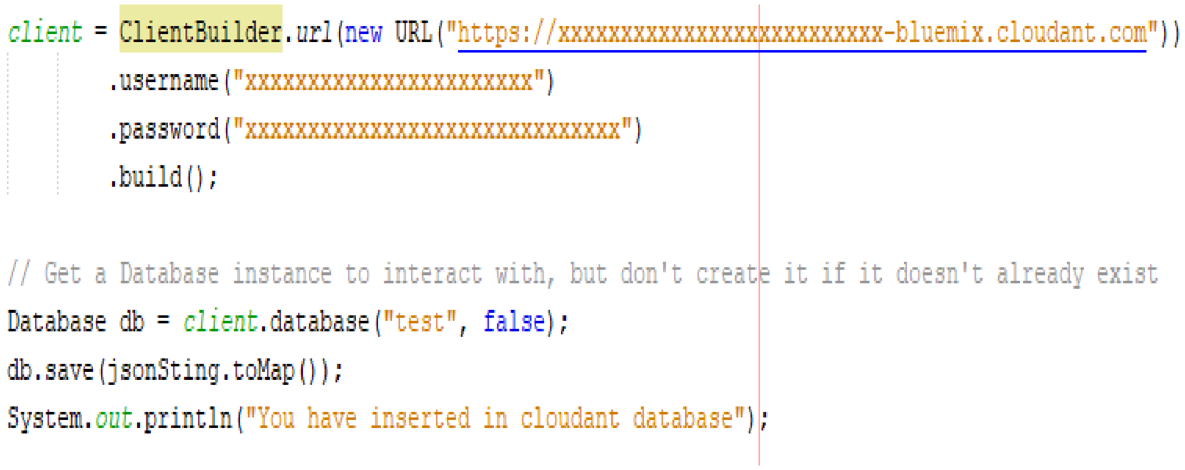
The data is read in JSON format over the serial interface over a synchronised serial even as shown in the code-snippet of Figure 4.3.



```
if (oEvent.getEventType() == SerialPortEvent.DATA_AVAILABLE) {
    try {
        String inputLine = null;
        if (input.ready()) {
            inputLine = input.readLine();
            System.out.println(inputLine);

            JSONObject readings = (new JSONObject(inputLine));
            Double wind_speed = readings.getDouble("WGmph");
            Double wind_dir = readings.getDouble("WGdir");
            Double humidity = readings.getDouble("Hum");
            Double temperature = readings.getDouble("Temp");
            Double rain_in = readings.getDouble("RainIn");
            Double pressure = readings.getDouble("Pre");
            Double light_lvl = readings.getDouble("LLvl");
```

Figure 4.3: Code Snippet for serial data acquisition

Once the data is read from the streaming serial interface, the connection to the IBM Cloudant NoSQL database is established using the codes and credentials as shown in Figure 4.4.

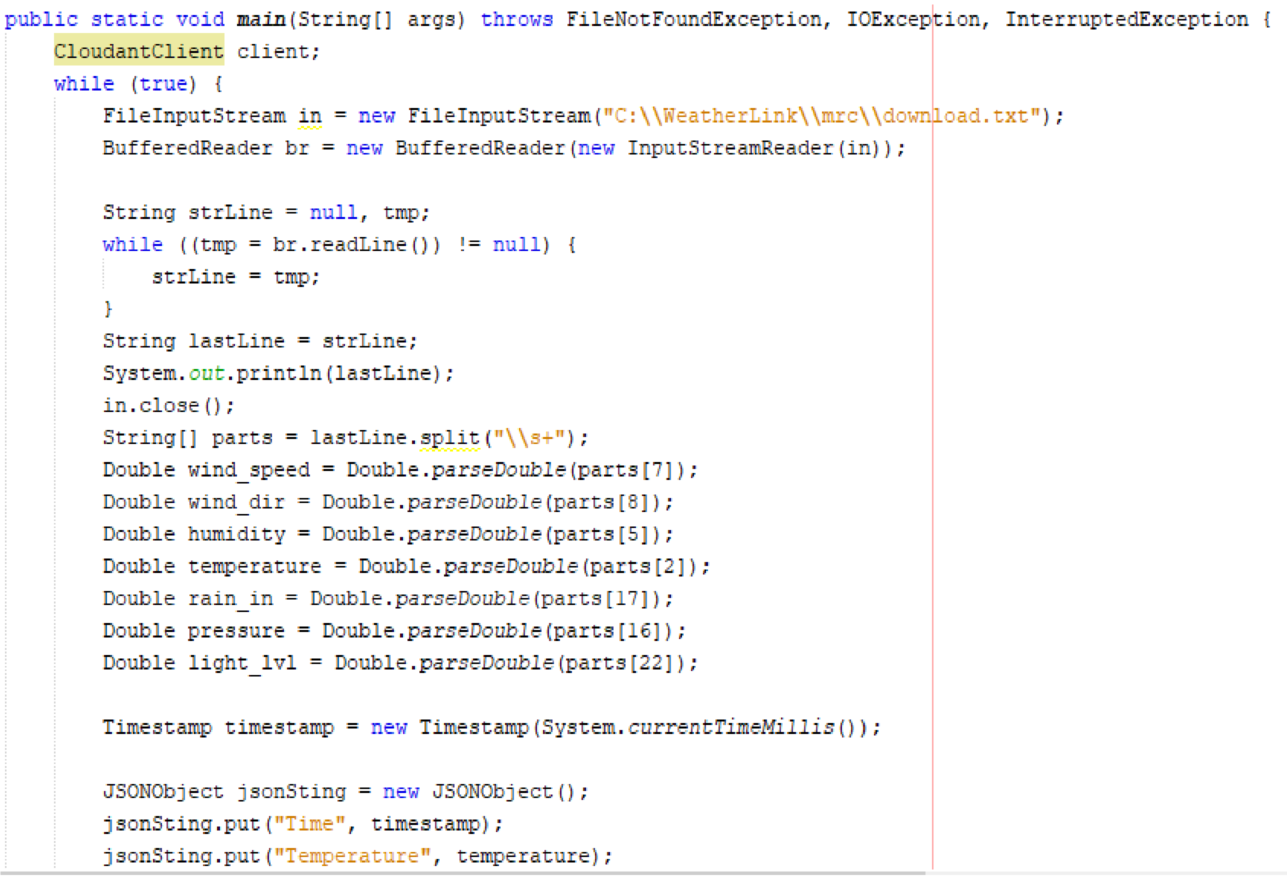


```
client = ClientBuilder.url(new URL("https://xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx-bluemix.cloudant.com"))
        .username("xxxxxxxxxxxxxxxxxxxxxxx")
        .password("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx")
        .build();

// Get a Database instance to interact with, but don't create it if it doesn't already exist
Database db = client.database("test", false);
db.save(jsonSting.toMap());
System.out.println("You have inserted in cloudant database");
```

Figure 4.4: Code Snippet for connection to Cloudant Database

4.2 Data capture codes for Davis Pro Vantage 2-based system

The data is read in JSON format from the text file stored on the local server through the data logging device as shown in the code-snippet of Figure 4.5. The codes for transferring the data to the Cloudant database are the same as shown in Figure 4.4.

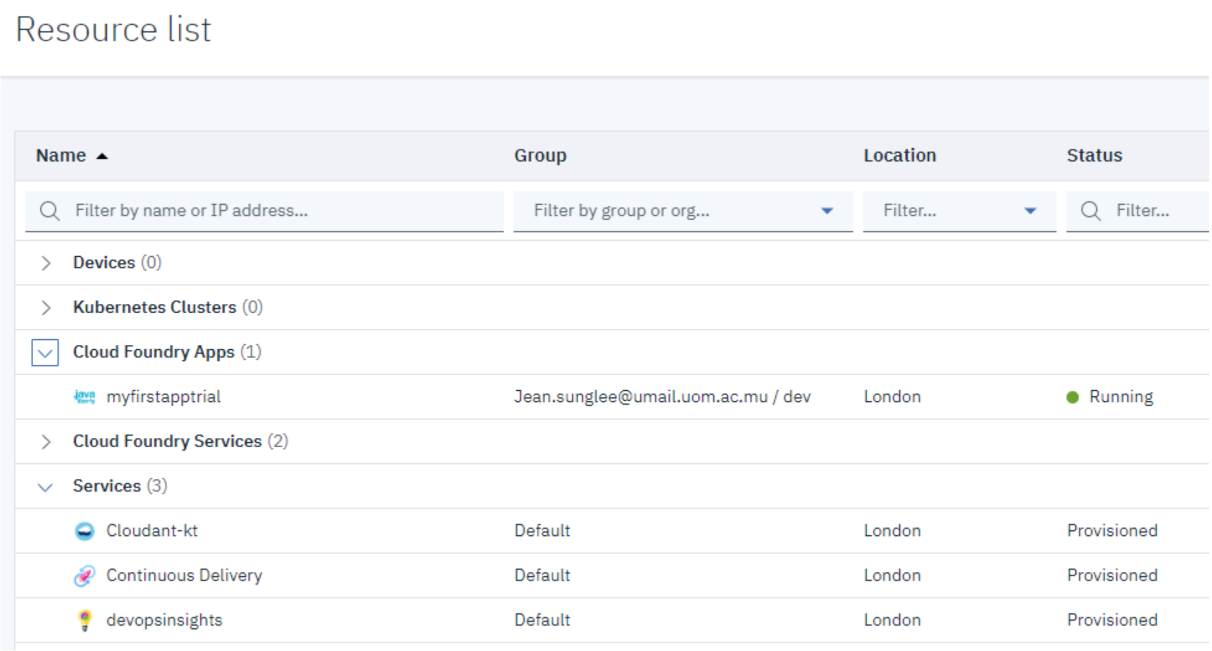


```
public static void main(String[] args) throws FileNotFoundException, IOException, InterruptedException {
    CloudantClient client;
    while (true) {
        FileInputStream in = new FileInputStream("C:\\WeatherLink\\mrc\\download.txt");
        BufferedReader br = new BufferedReader(new InputStreamReader(in));

        String strLine = null, tmp;
        while ((tmp = br.readLine()) != null) {
            strLine = tmp;
        }
        String lastLine = strLine;
        System.out.println(lastLine);
        in.close();
        String[] parts = lastLine.split("\\s+");
        Double wind_speed = Double.parseDouble(parts[7]);
        Double wind_dir = Double.parseDouble(parts[8]);
        Double humidity = Double.parseDouble(parts[5]);
        Double temperature = Double.parseDouble(parts[2]);
        Double rain_in = Double.parseDouble(parts[17]);
        Double pressure = Double.parseDouble(parts[16]);
        Double light_lvl = Double.parseDouble(parts[22]);

        Timestamp timestamp = new Timestamp(System.currentTimeMillis());

        JSONObject jsonSting = new JSONObject();
        jsonSting.put("Time", timestamp);
        jsonSting.put("Temperature", temperature);
```

Figure 4.5: Part of Code Snippet for data acquisition from Davis Vantage Pro 2

4.3 IBM Cloud Services

The resource list for the services used on the IBM cloud platform are as shown in Figure 4.6.



Resource list

Name ▲	Group	Location	Status
Filter by name or IP address...	Filter by group or org... ▾	Filter... ▾	Filter...
› **Devices** (0)			
› **Kubernetes Clusters** (0)			
⌄ **Cloud Foundry Apps** (1)			
myfirstapptrial	Jean.sunglee@umail.uom.ac.mu / dev	London	● Running
› **Cloud Foundry Services** (2)			
⌄ **Services** (3)			
Cloudant-kt	Default	London	Provisioned
Continuous Delivery	Default	London	Provisioned
devopsinsights	Default	London	Provisioned

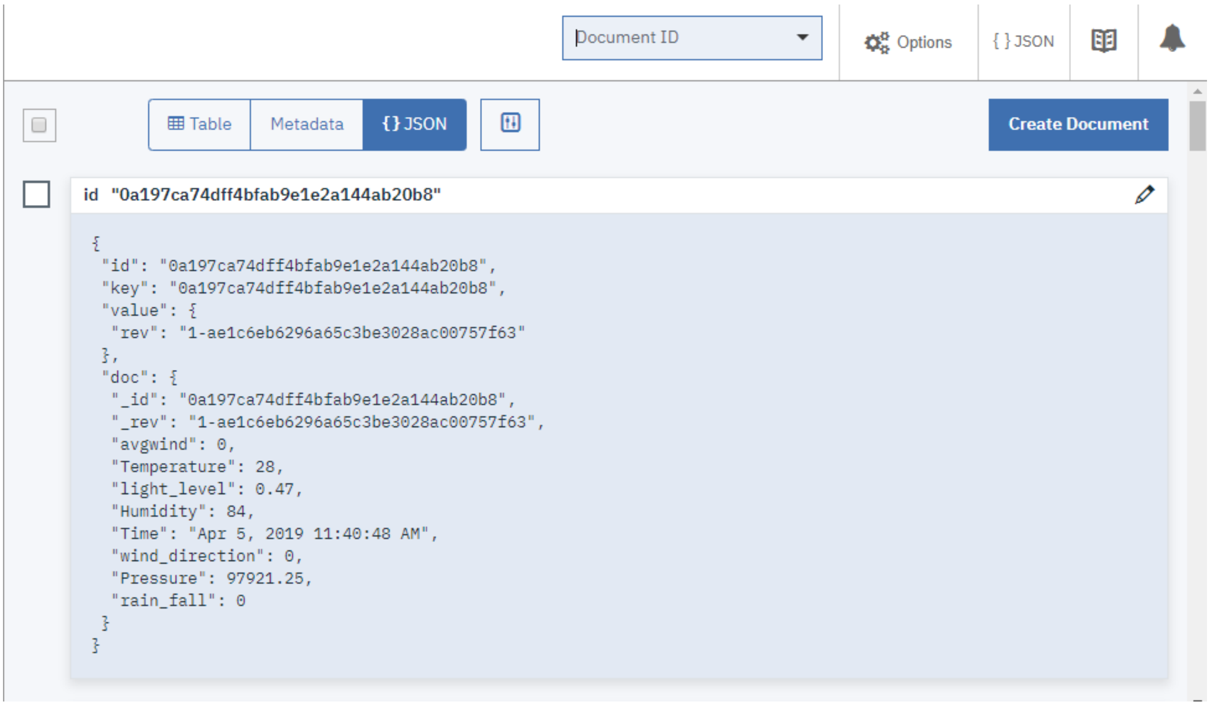
Figure 4.6: resource list for services on IBM Cloud platform

The services used are:

- The Cloud Foundry Apps for hosting the webpage,
- The Cloudant-kt for unstructured data storage,
- The Continuous Delivery service for real-time storage and retrieval,
- The Devops Insights service for hosting the JAVA-based prediction algorithm.

4.4 Architecture for IBM Cloudant Database

The IBM Cloudant database is a NoSQL database which allows for the handling of Big Data when the phase of mass data collection is reached. The parameters are stored in documents in a database in contrast to fields in tables for structured data. The structure of a sample data stored is shown in Figure 4.7. All parameters are stored in documents with an ID associated to each document.



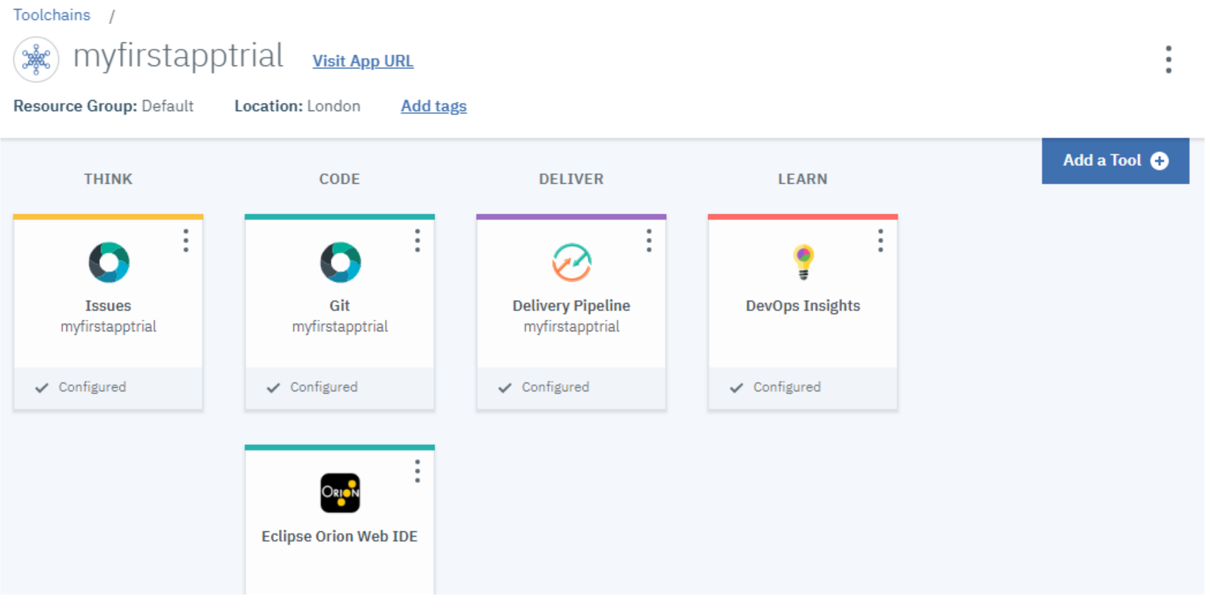
id "0a197ca74dff4bfab9e1e2a144ab20b8"

```
{
 "id": "0a197ca74dff4bfab9e1e2a144ab20b8",
 "key": "0a197ca74dff4bfab9e1e2a144ab20b8",
 "value": {
  "rev": "1-ae1c6eb6296a65c3be3028ac00757f63"
 },
 "doc": {
  "_id": "0a197ca74dff4bfab9e1e2a144ab20b8",
  "_rev": "1-ae1c6eb6296a65c3be3028ac00757f63",
  "avgwind": 0,
  "Temperature": 28,
  "light_level": 0.47,
  "Humidity": 84,
  "Time": "Apr 5, 2019 11:40:48 AM",
  "wind_direction": 0,
  "Pressure": 97921.25,
  "rain_fall": 0
 }
}
```

Figure 4.7: Sample Data stored on Cloudant database.

4.5 Architecture for IBM Devops Insights

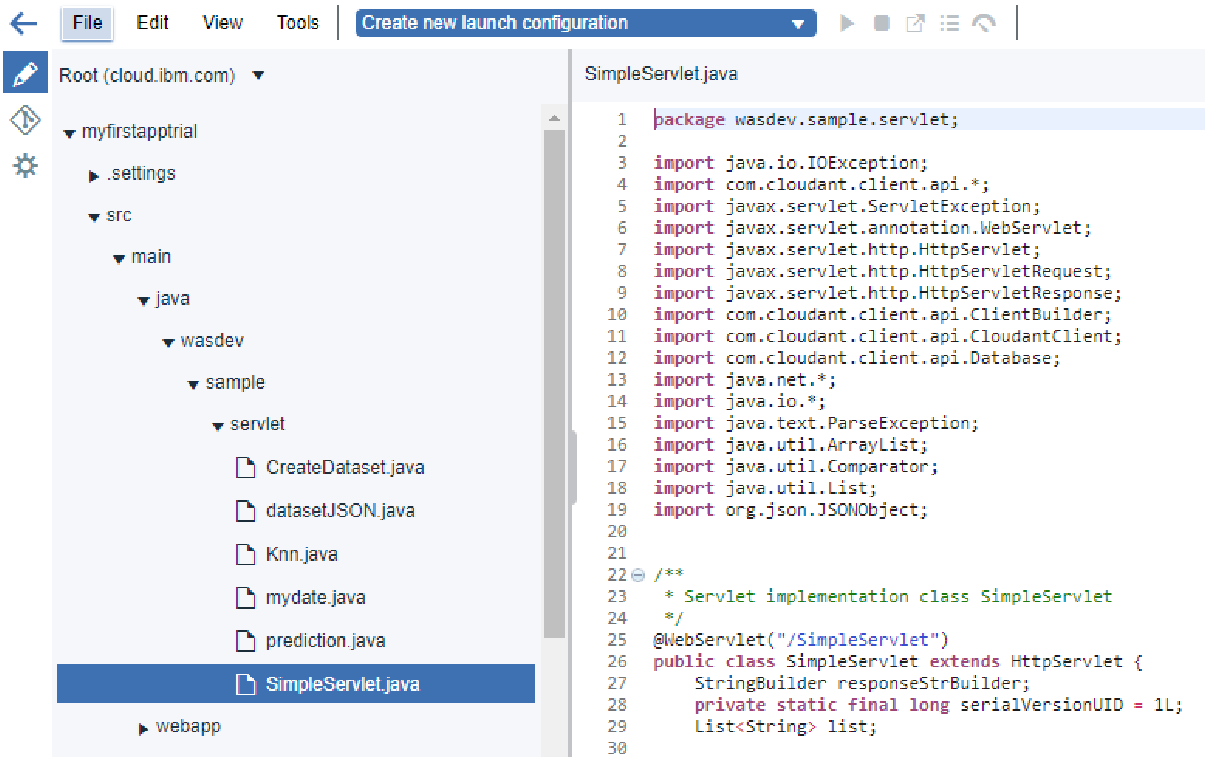
The IBM Devops Insights has been used for the deployment of the JAVA-based prediction algorithm by making use of the Tool Chain facility and integrating the Eclipse Orion Web utility for coding, and Git for change management and deployment through the Delivery pipeline as shown in Figure 4.8.



Resource Group: Default **Location:** London

Figure 4.8: Toolchains on IBM Cloud [67].

The Eclipse IDE provides an almost similar experience in terms of Class / Method creation and coding through a web interface. A sample for the weather prediction algorithm is as shown in Figure 4.9.

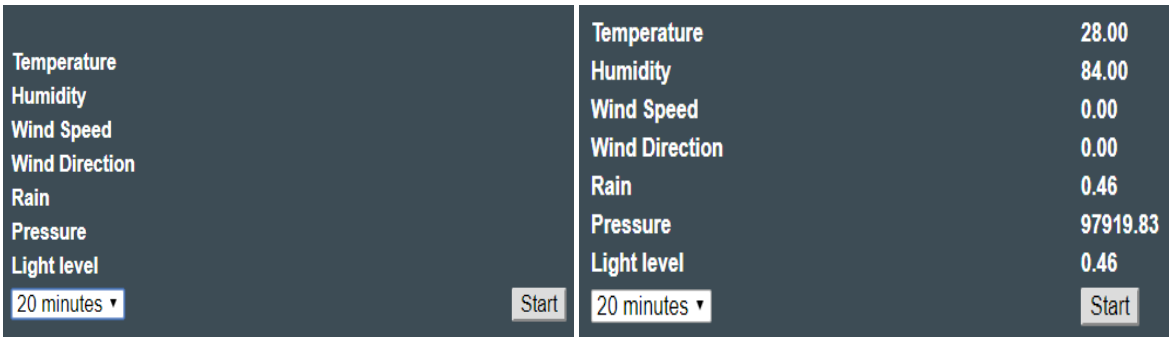


```
package wasdev.sample.servlet;

import java.io.IOException;
import com.cloudant.client.api.*;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.cloudant.client.api.ClientBuilder;
import com.cloudant.client.api.CloudantClient;
import com.cloudant.client.api.Database;
import java.net.*;
import java.io.*;
import java.text.ParseException;
import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;
import org.json.JSONObject;


/**
 * Servlet implementation class SimpleServlet
 */
@WebServlet("/SimpleServlet")
public class SimpleServlet extends HttpServlet {
    StringBuilder responseStrBuilder;
    private static final long serialVersionUID = 1L;
    List<String> list;

```

Figure 4.9: Sample code on IBM Devops Insights using the Eclipse Orion Web IDE.

4.6 Web and Mobile Application for Weather prediction

The web application is hosted at the link: https://myfirstapptrial.eu-gb.mybluemix.net/ and a sample is shown in Figure 4.10.

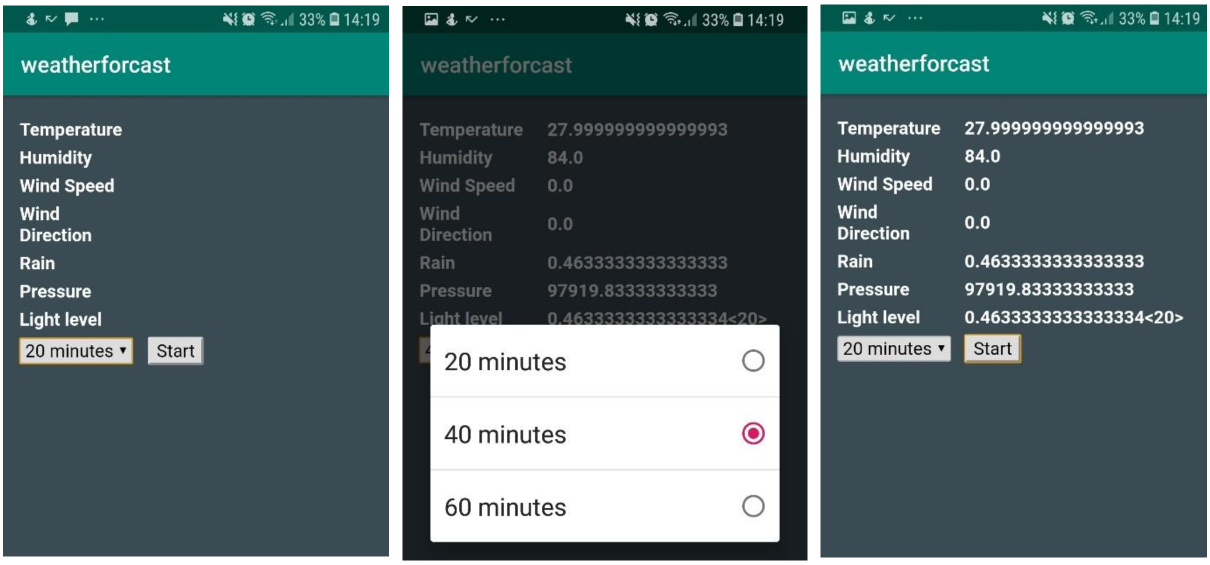


Temperature
Humidity
Wind Speed
Wind Direction
Rain
Pressure
Light level

Temperature	28.00
Humidity	84.00
Wind Speed	0.00
Wind Direction	0.00
Rain	0.46
Pressure	97919.83
Light level	0.46

Figure 4.10: Sample of web-based application

The sample mobile application is as shown in Figure 4.11. The parameter selection of 20, 40, or 60 minutes represents the amount of time after which the prediction is being sought. The selected value is then fed to the algorithm running on the cloud which performs the computations and sends back the predicted values.



Temperature
Humidity
Wind Speed
Wind Direction
Rain
Pressure
Light level
20 minutes ▾

○ 20 minutes
● 40 minutes
○ 60 minutes

Temperature	27.999999999999993
Humidity	84.0
Wind Speed	0.0
Wind Direction	0.0
Rain	0.46333333333333333
Pressure	97919.83333333333
Light level	0.46333333333333334<20>

20 minutes ▾

Figure 4.11: Sample of mobile application

CHAPTER 5

RESULTS AND DISCUSSIONS

In this chapter the results obtained for the tests conducted on the UoM campus at Reduit, Terre Rouge and Vacoas are given and analysed.

The first section gives an overview of the weather data captured at the three different locations and the second section gives a detailed analysis of the performance of the algorithms tested.

5.1 Data captured in three different locations with the weather forecasting system

5.1.1 Data Captured on UoM Campus at Reduit

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. The setup for taking the results is shown in Figure 5.1.

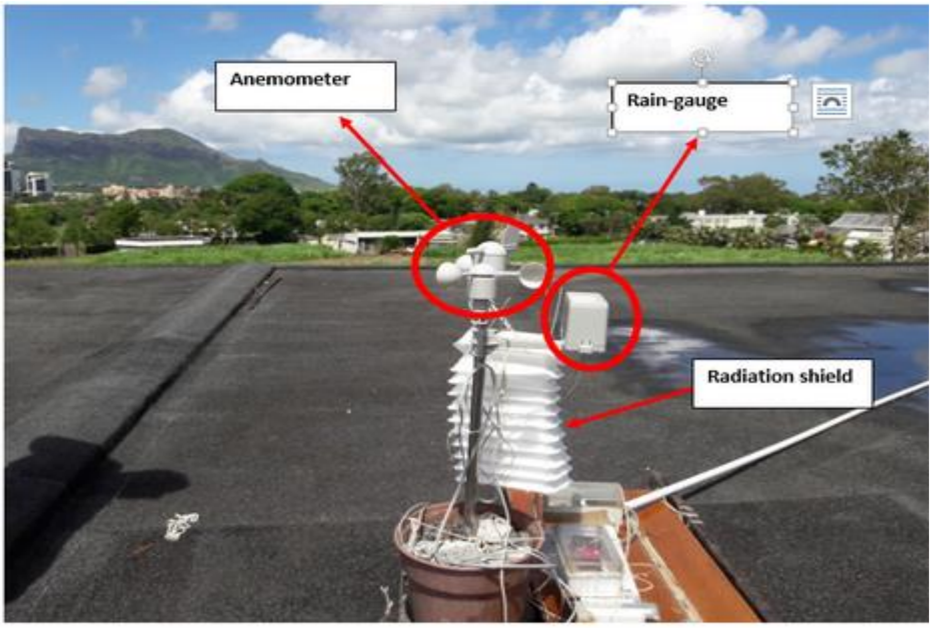


Figure 5.1: Setup used at the UoM Campus.

The graphs in Figures 5.2 to 5.8 show the actual values for all parameters (Temperature, Humidity, Wind Speed, Wind Direction, Rain, Pressure and Light intensity) recorded on 23 February 2018 from 0900 to 1600 at a rate of 12 values per minute.

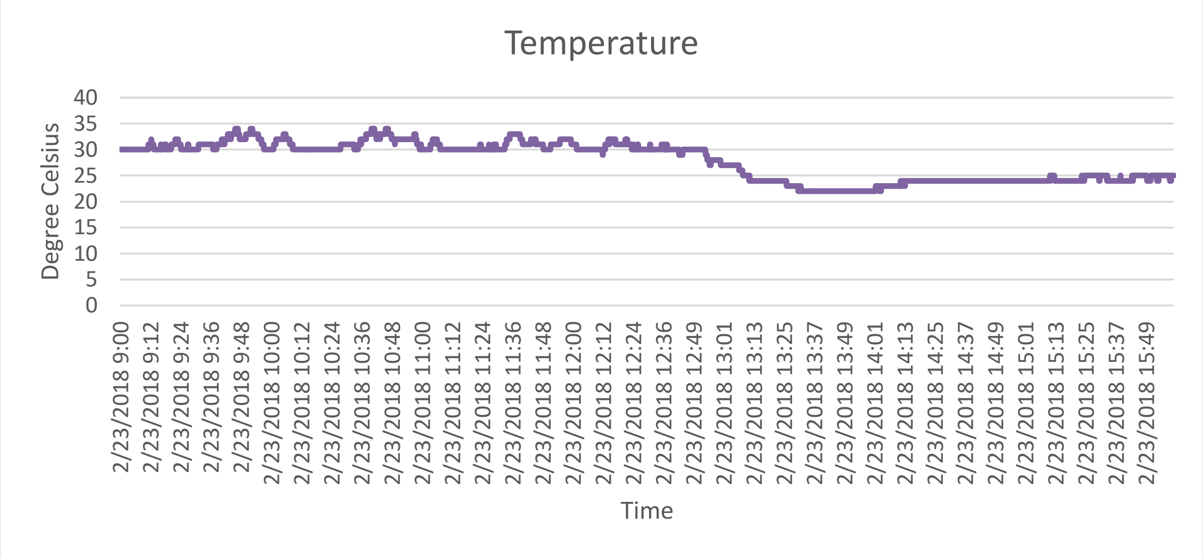


Figure 5.2: Temperature readings for 23 February 2018

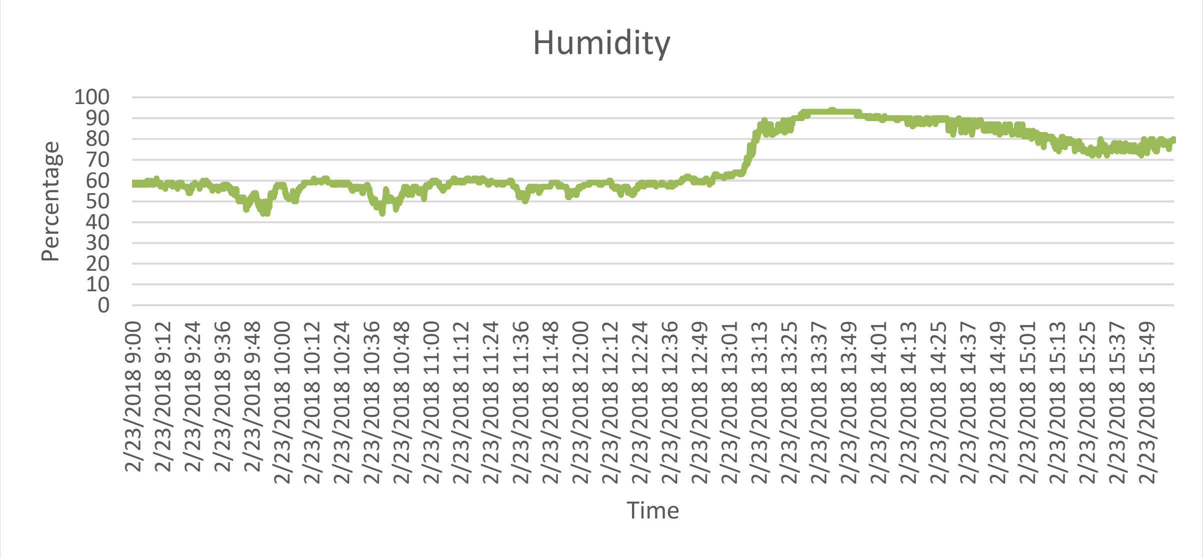


Figure 5.3: Humidity readings on the 23 February 2018

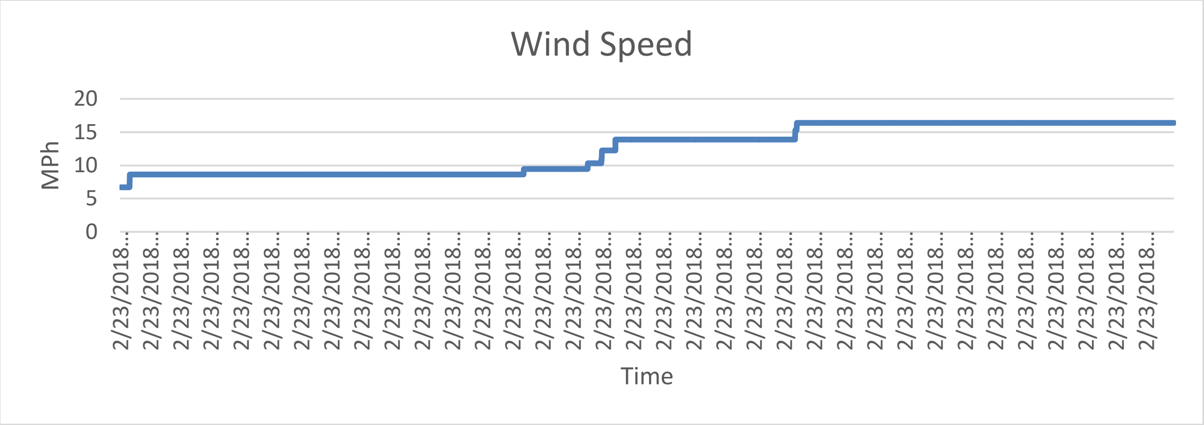


Figure 5.4: Wind Speed readings on the 23 February 2018

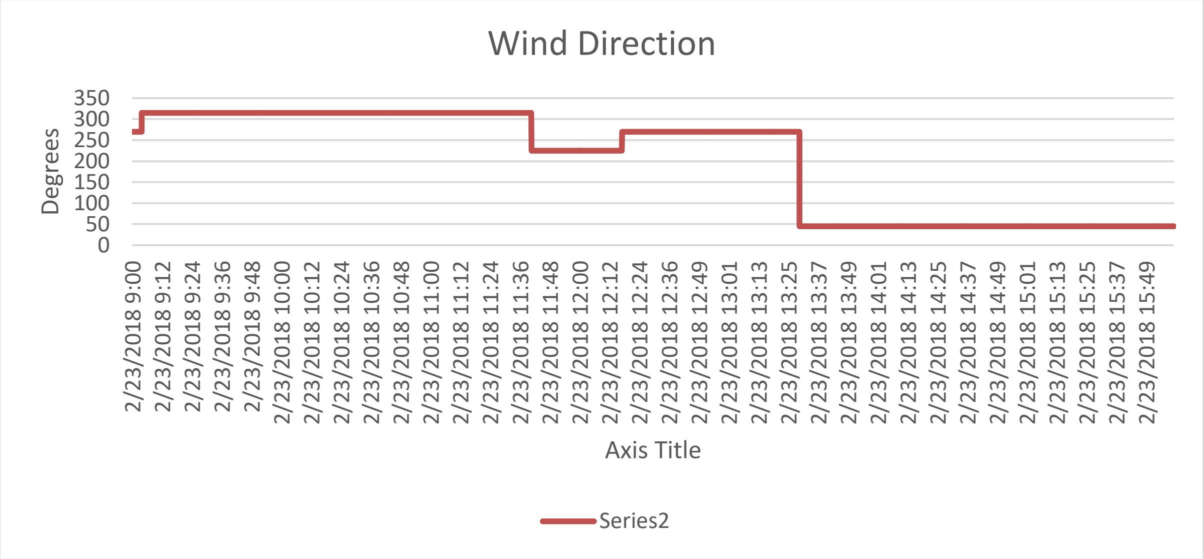


Figure 5.5: Wind Direction readings on the 23 February 2018

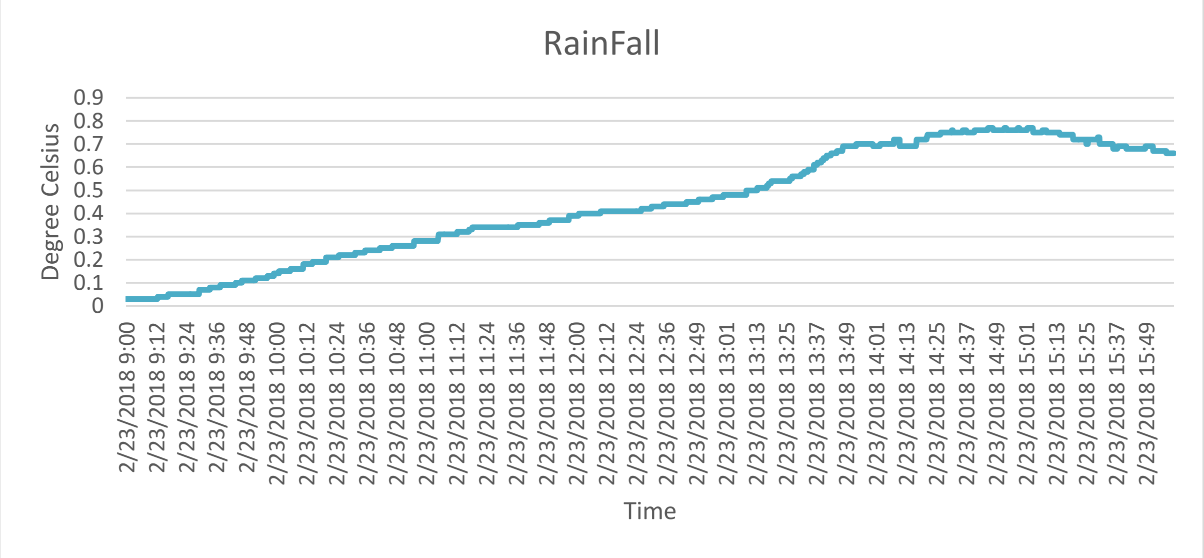


Figure 5.6: Rain readings on the 23 February 2018

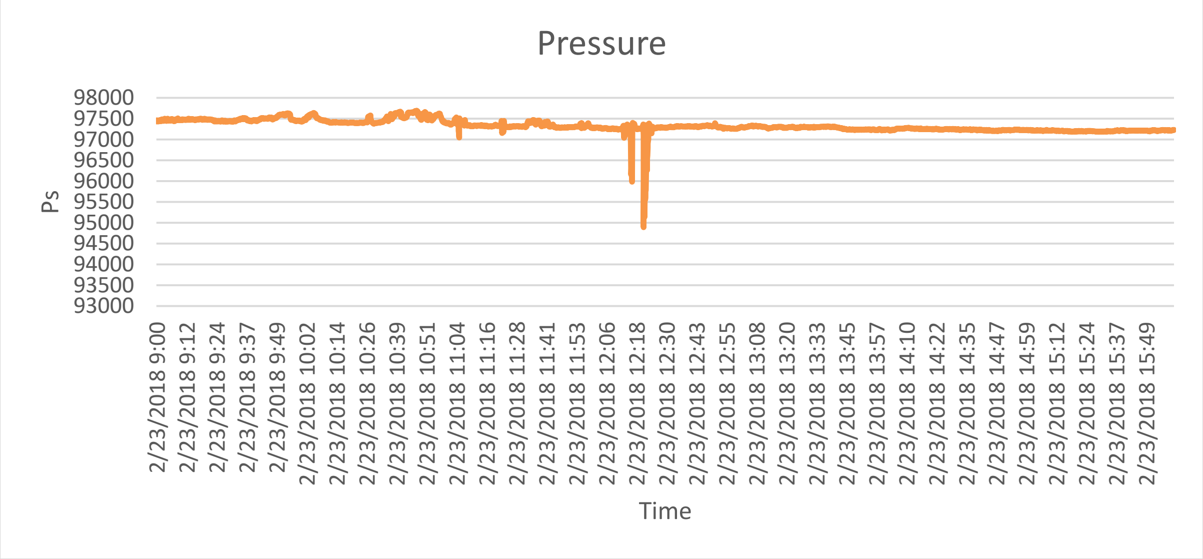


Figure 5.7: Pressure readings on the 23 February 2018

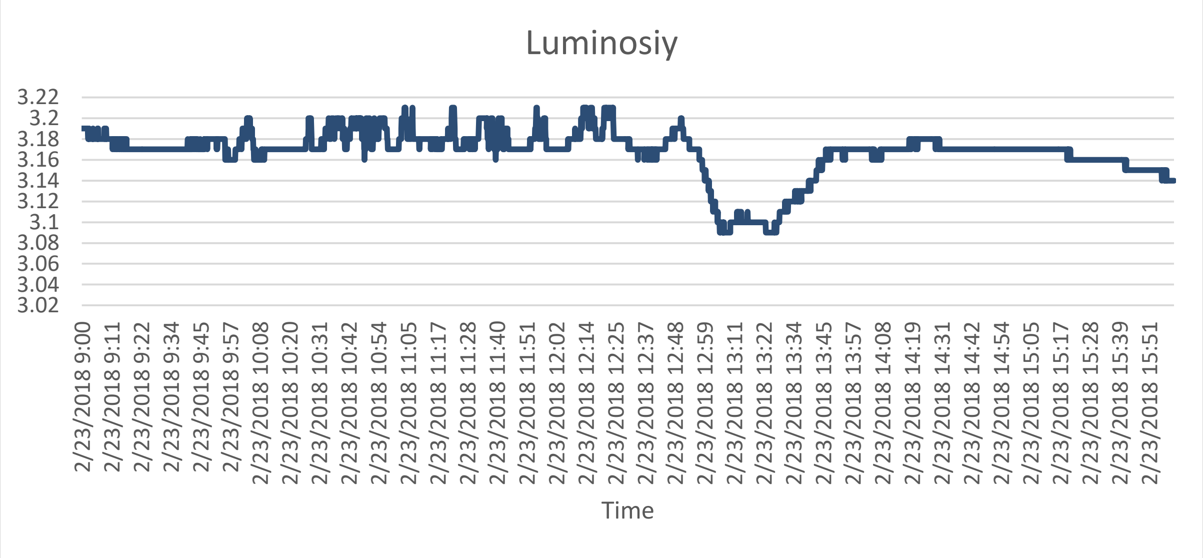


Figure 5.8: Light intensity readings on the 23 February 2018

5.1.2 Data Captured at Terre Rouge

The tests were conducted at Terre Rouge from 10th February 2019 to 21st February 2019 with data collected from 0900 to 1600 on each day. The setup for taking the results is shown in Figure 5.9.

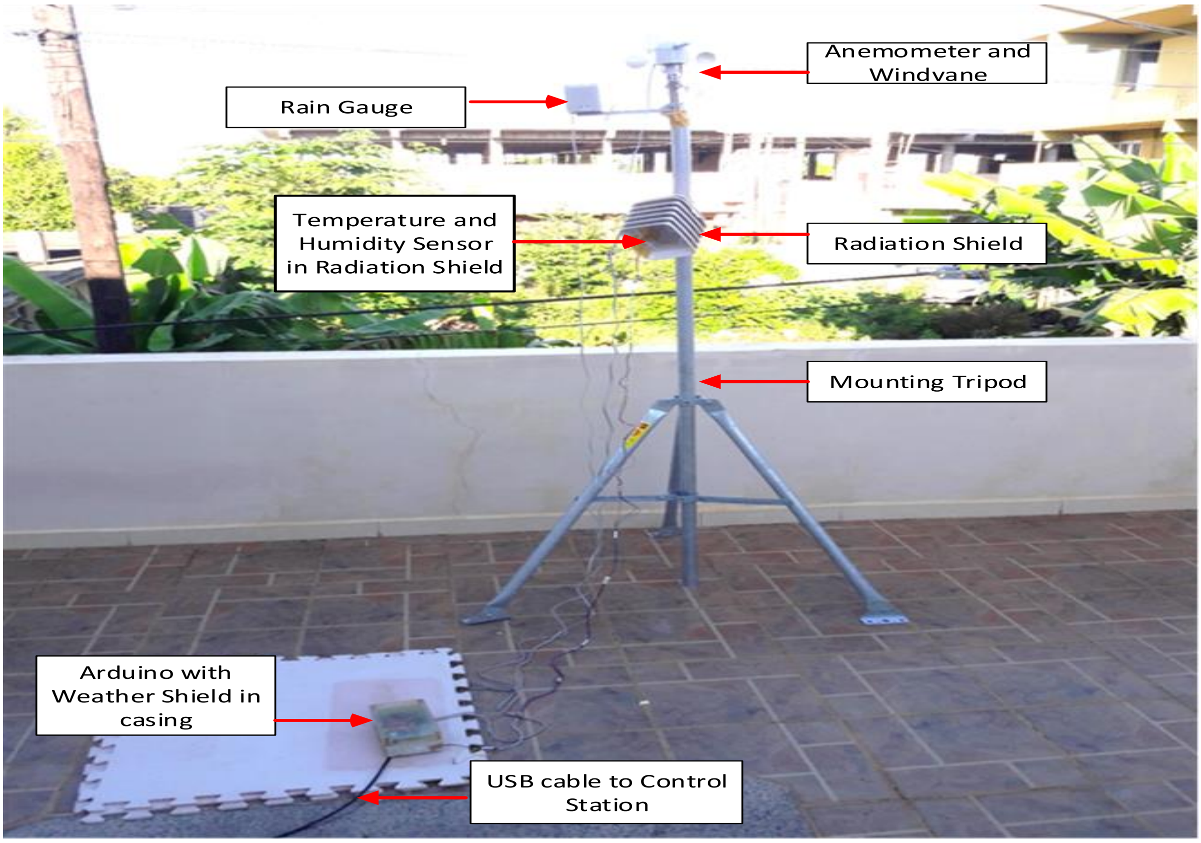

Anemometer and Windvane
Rain Gauge
Temperature and Humidity Sensor in Radiation Shield
Radiation Shield
Mounting Tripod
Arduino with Weather Shield in casing
USB cable to Control Station

Figure 5.10: Setup used at Terre Rouge.

The graphs in Figures 5.11 to 5.17 show the actual values for all parameters (Temperature, Humidity, Wind Speed, Wind Direction, Rain, Pressure and Light intensity) recorded on 17th February 2019 from 0900 to 1600 at a rate of 12 values per minute.

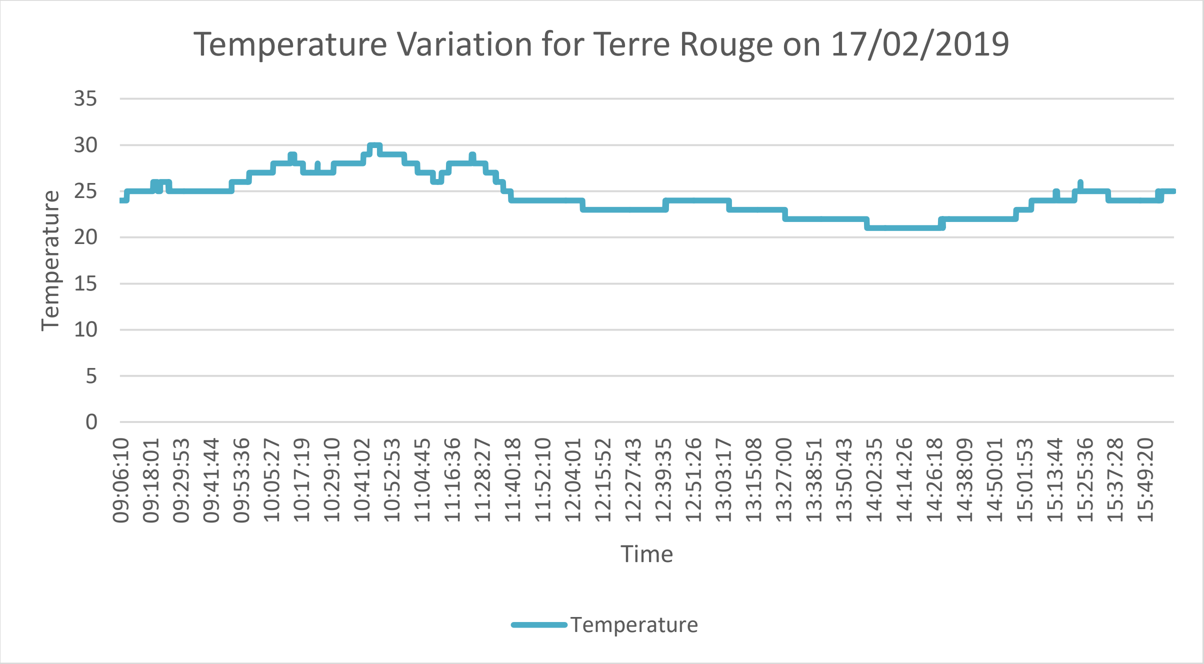


Figure 5.11: Temperature readings for 17 February 2019

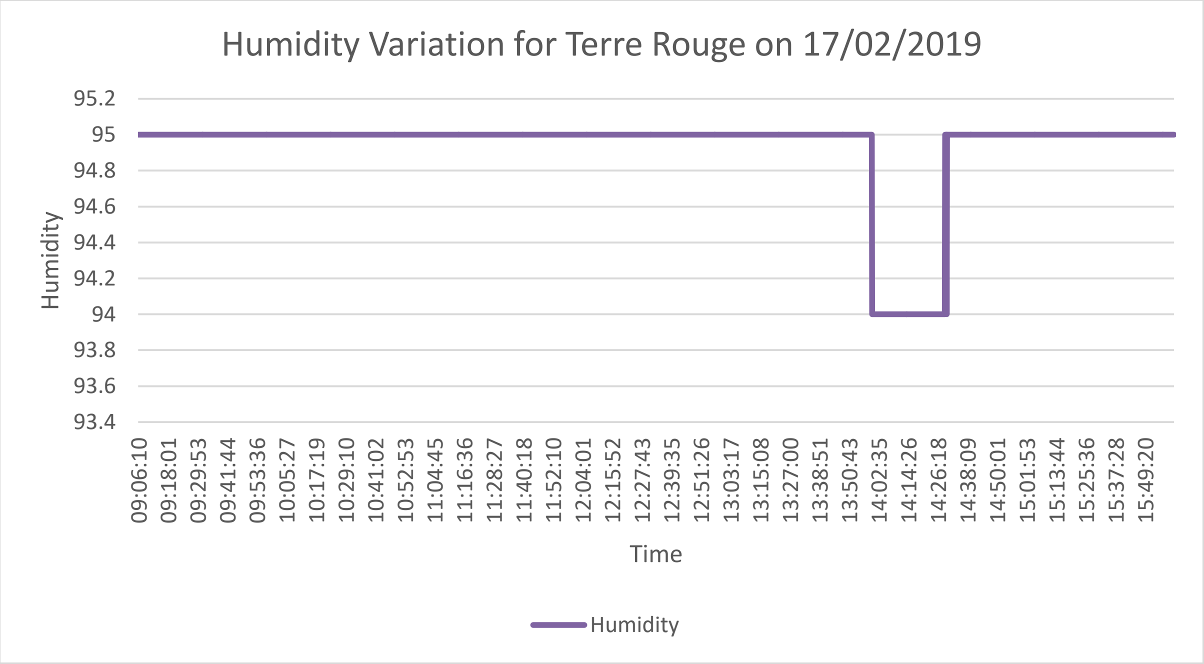


Figure 5.12: Humidity readings for 17 February 2019

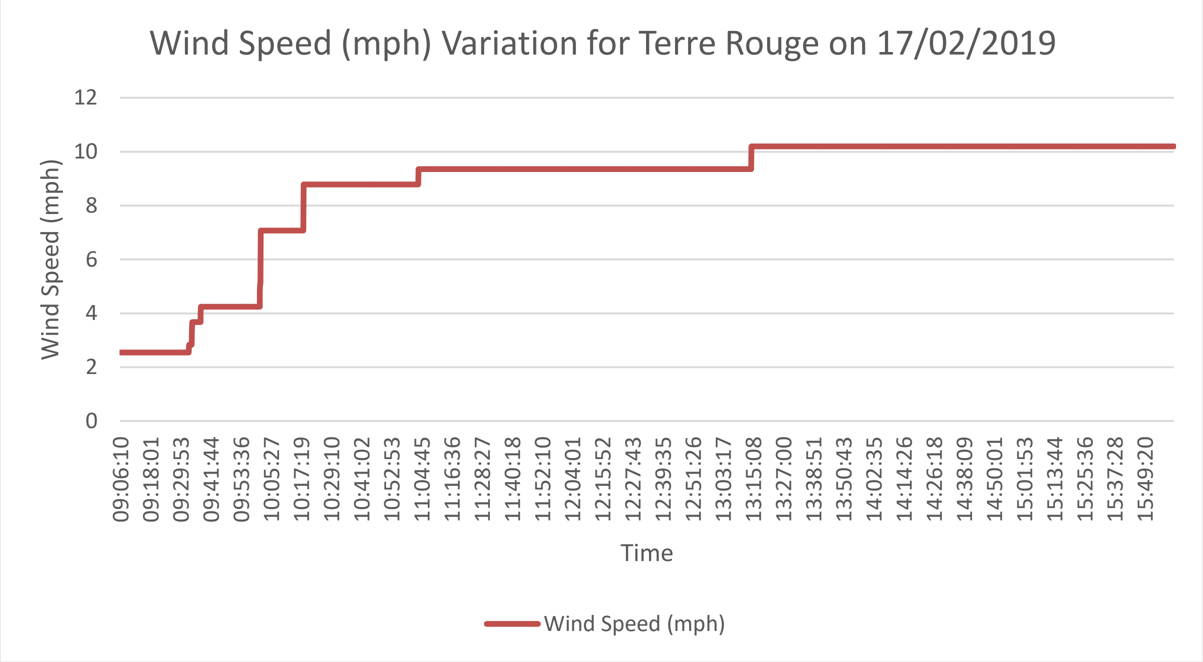


Figure 5.13: Wind speed readings for 17 February 2019

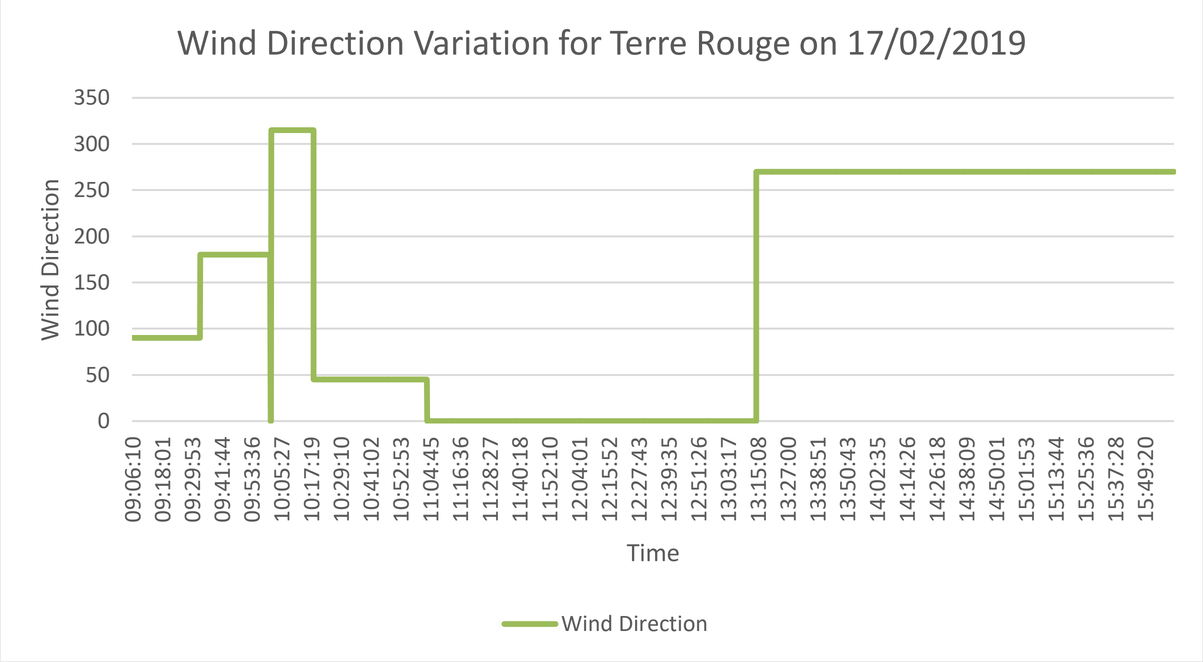


Figure 5.14: Wind direction readings for 17 February 2019

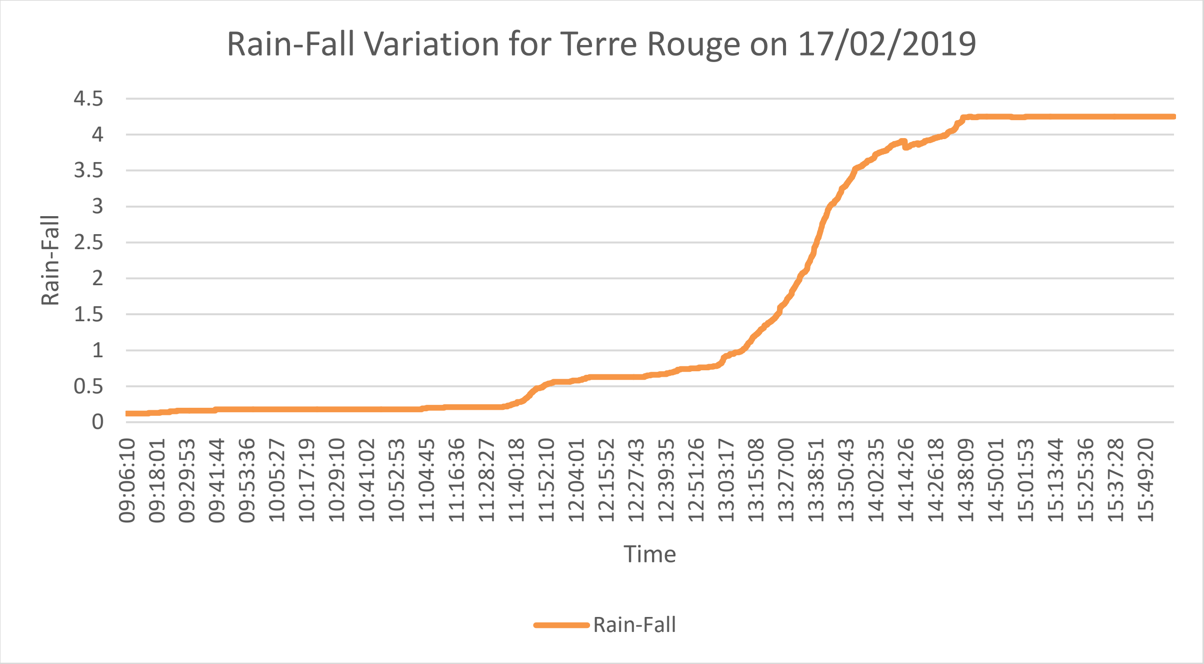


Figure 5.15: Rainfall readings for 17 February 2019

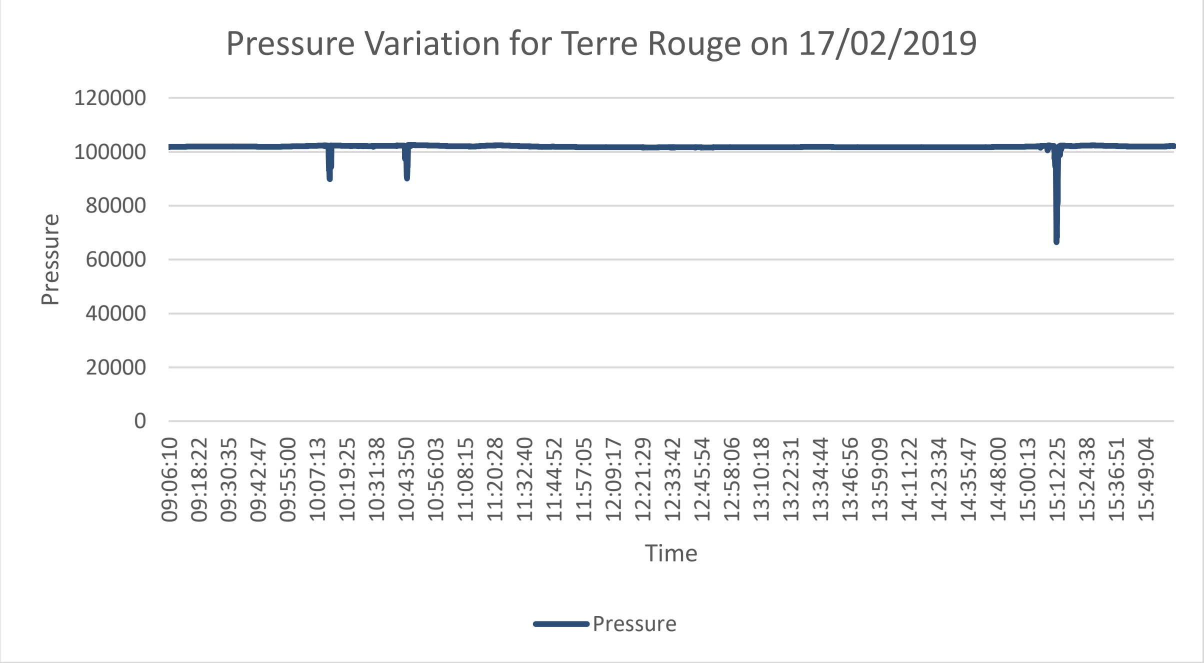


Figure 5.16: Pressure readings for 17 February 2019

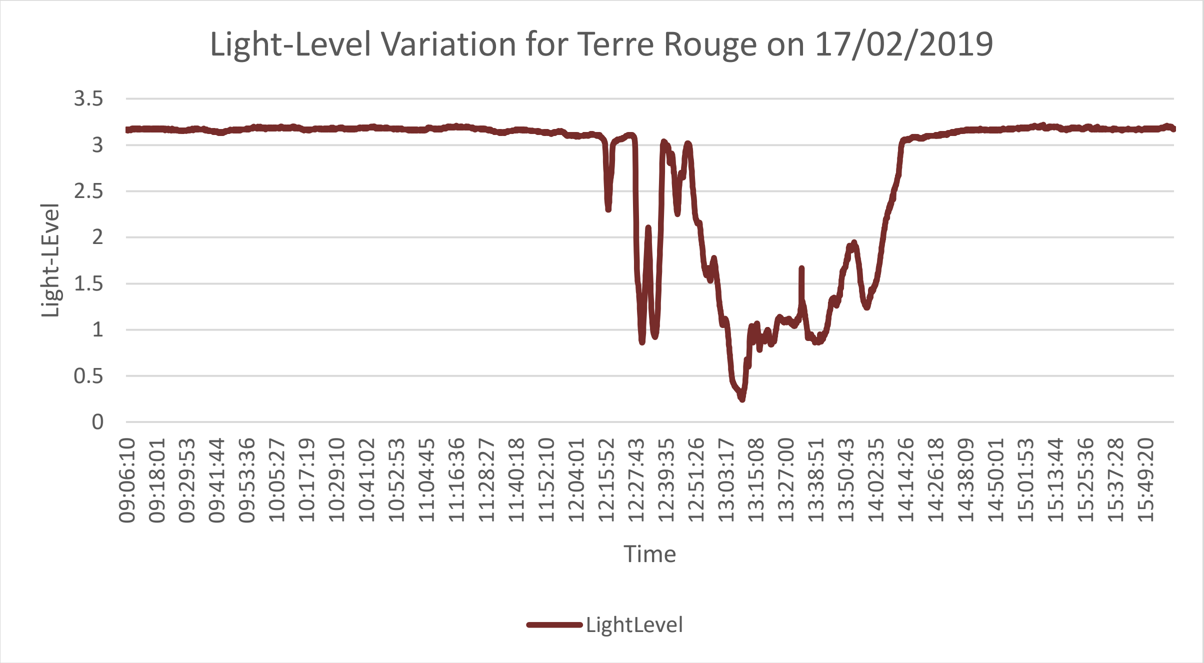


Figure 5.17: Light intensity readings for 17 February 2019

5.1.3 Data Captured at Vacoas

The tests were conducted at Vacoas from 01/March/2019 to 10th /March /2019 with data collected from 0900 to 1600 on each day. The setup for taking the results is shown in Figure 5.18.

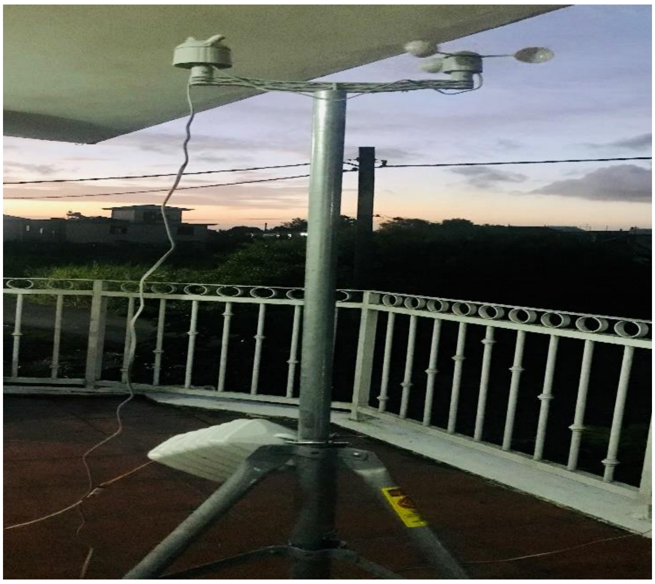


Figure 5.18: Set-up at Vacoas.

The graphs in Figures 5.19 to 5.20 show the actual values for all parameters (Temperature, Humidity, Wind Speed, Wind Direction, Rain, Pressure and Light intensity) recorded on 2nd March 2019 from 0900 to 1600 at a rate of 12 values per minute.

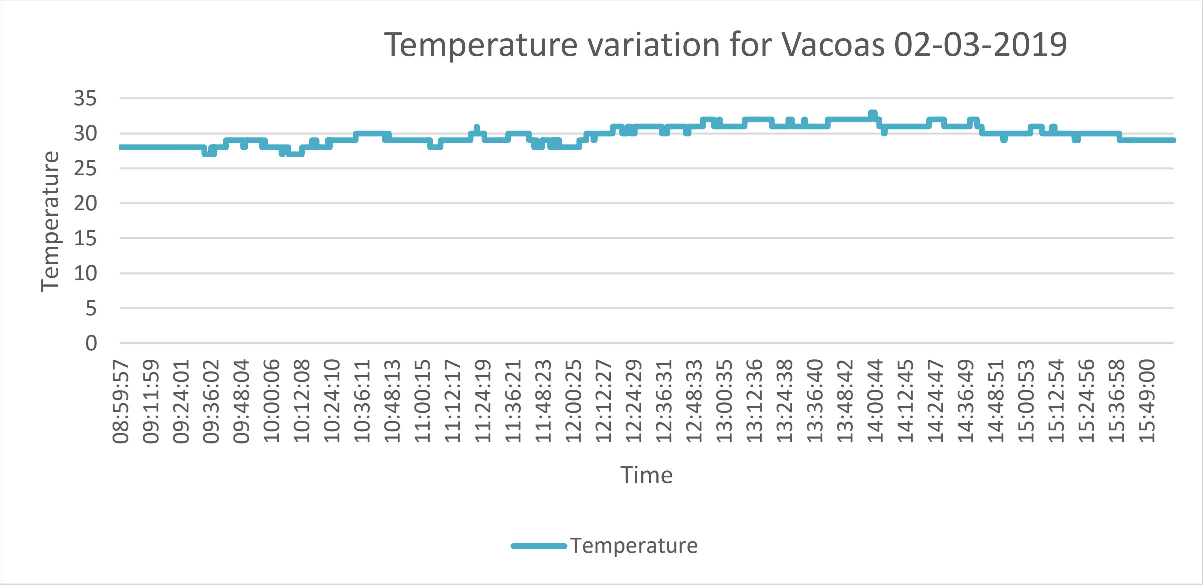


Figure 5.19: Temperature readings for 2nd March 2019

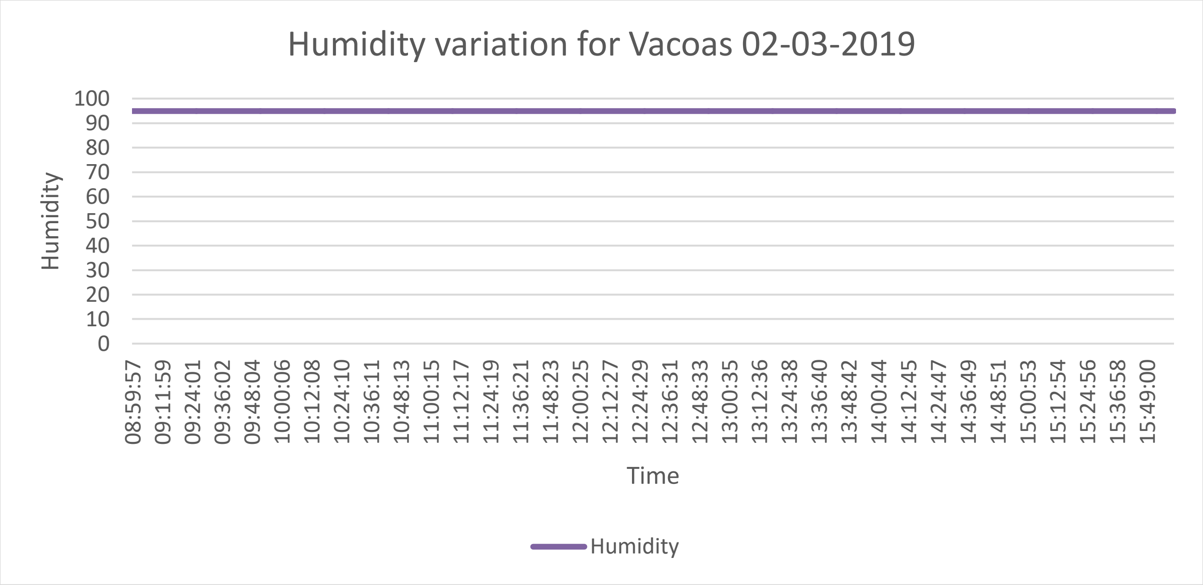


Figure 5.20: Humidity readings for 2nd March 2019

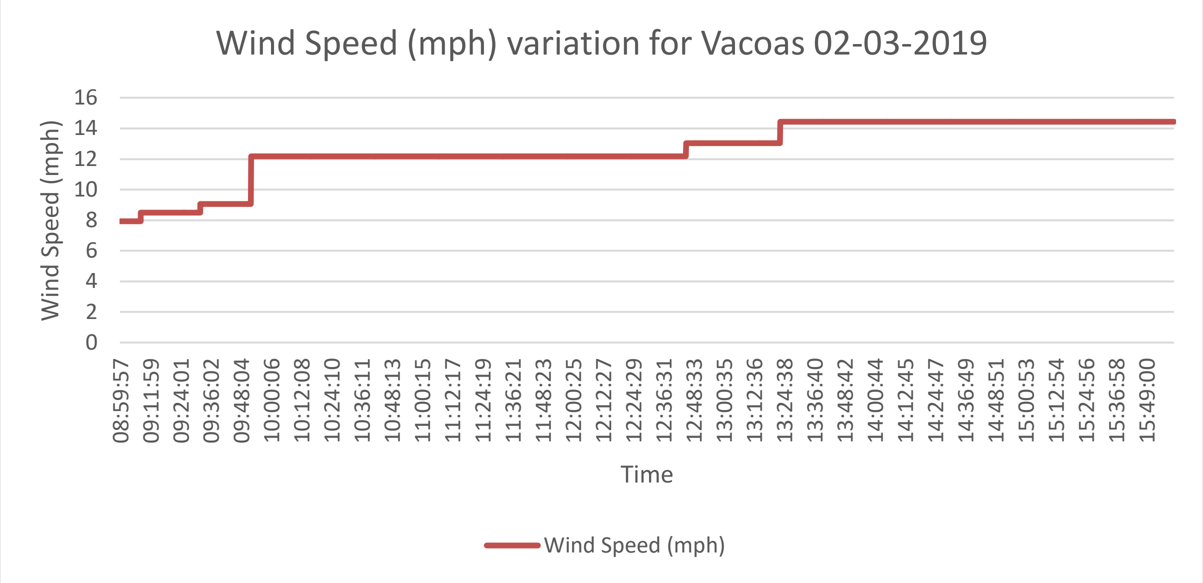


Figure 5.21: Wind speed readings for 2nd March 2019

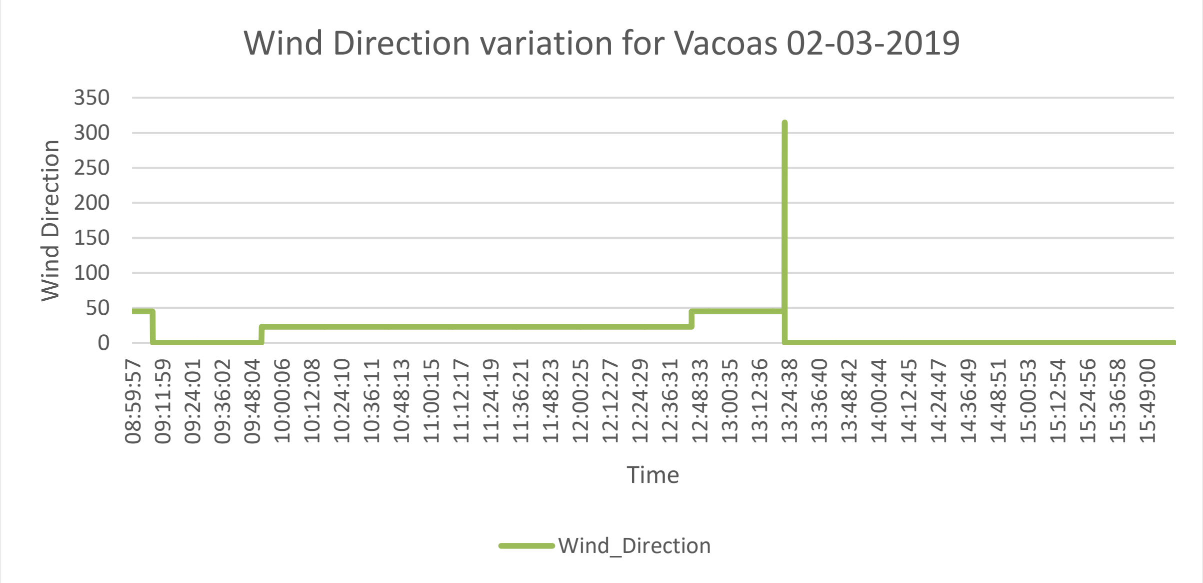


Figure 5.22: Wind direction readings for 2nd March 2019

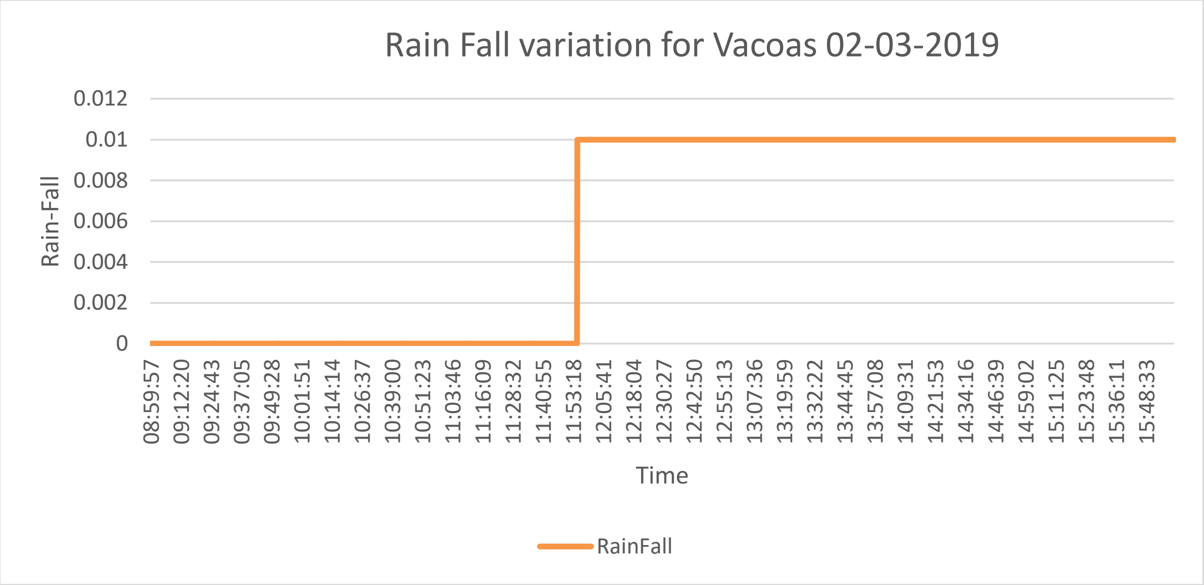


Figure 5.23: Rainfall readings for 2nd March 2019

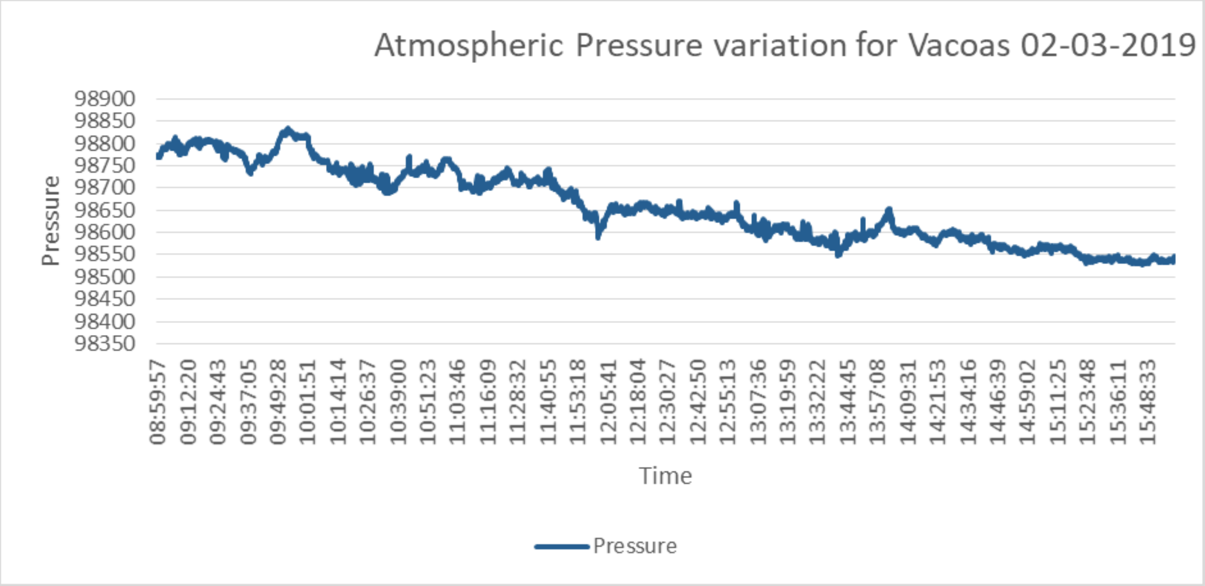


Figure 5.24: Pressure readings for 2nd March 2019

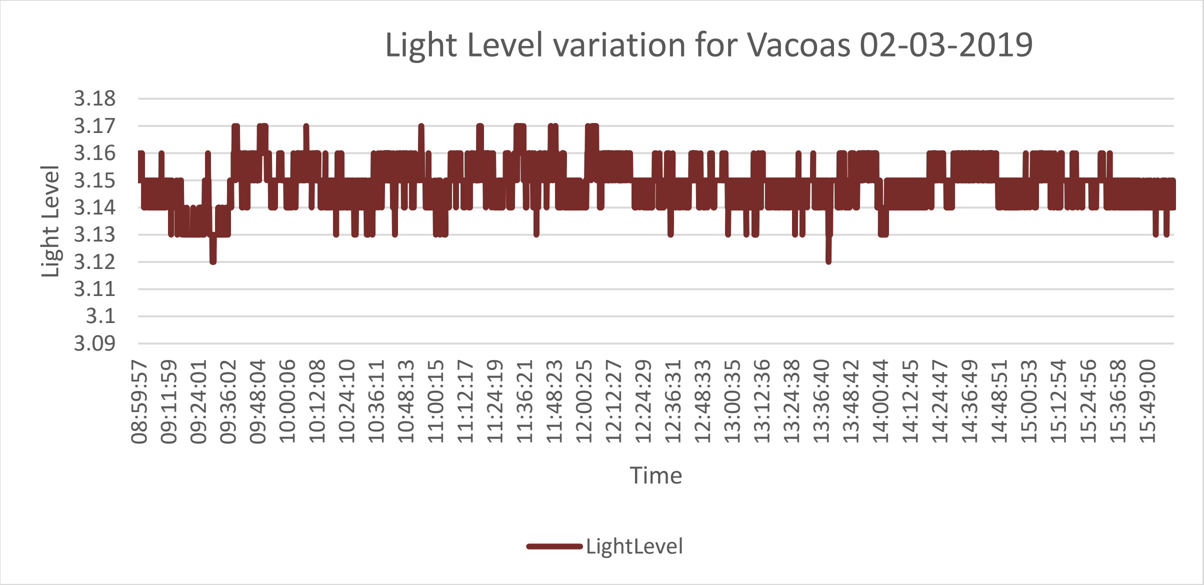


Figure 5.25: Light intensity readings for 2nd March 2019

5.2 Performance analysis of the Weather forecasting system

Using a subset of the data captured between 0900-1600, predictions were made between 1200 to 1600 by the following schemes:

1. KNN
2. Adaptive KNN
3. Multiple Linear Regression
4. Multiple Linear Regression 1
5. Multiple Linear Regression 2
6. Adaptive Multiple Linear Regression
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)

A window of 2 hours was used as a training dataset to predict at intervals of 20 minutes (P20), 40 minutes (P40) and 60 minutes (P60). Readings for the following parameters were obtained at a rate of 12 values per minute:

1. Temperature
2. Humidity
3. Wind Speed
4. Wind Direction
5. Rain
6. Pressure
7. Light intensity

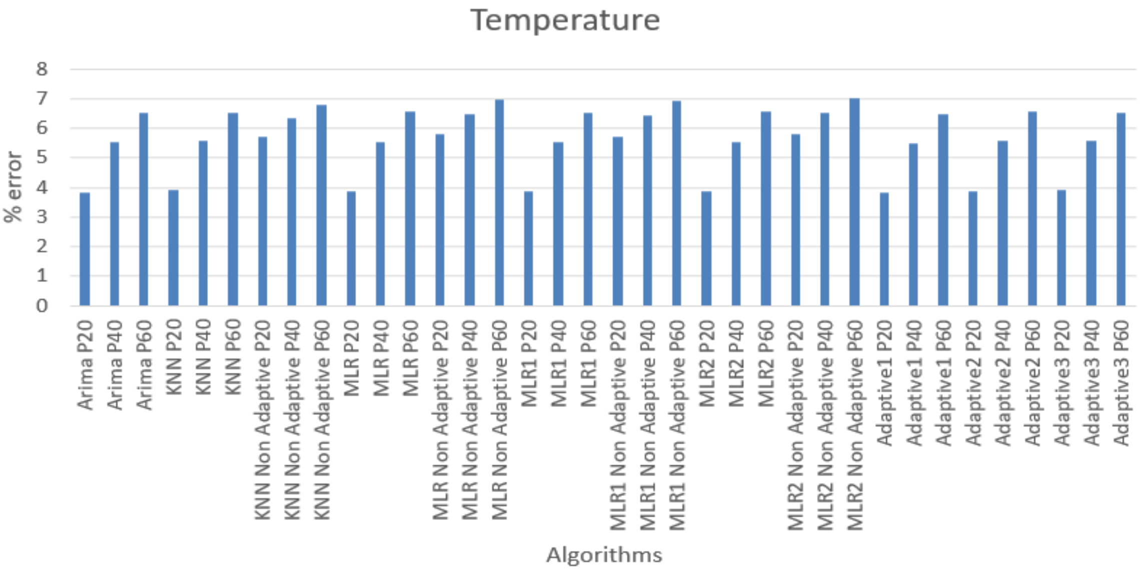
For any given parameter y, the average percentage error Ey between the actual values A_y recorded by the sensors and the predicted values V_y, was computed over the whole period of measurement as follows:

$$E_y = \frac{1}{N}\sum_{t=1}^{N}\frac{|A_x(t) - V_x|}{A_x(t)}x100 \tag{1}$$

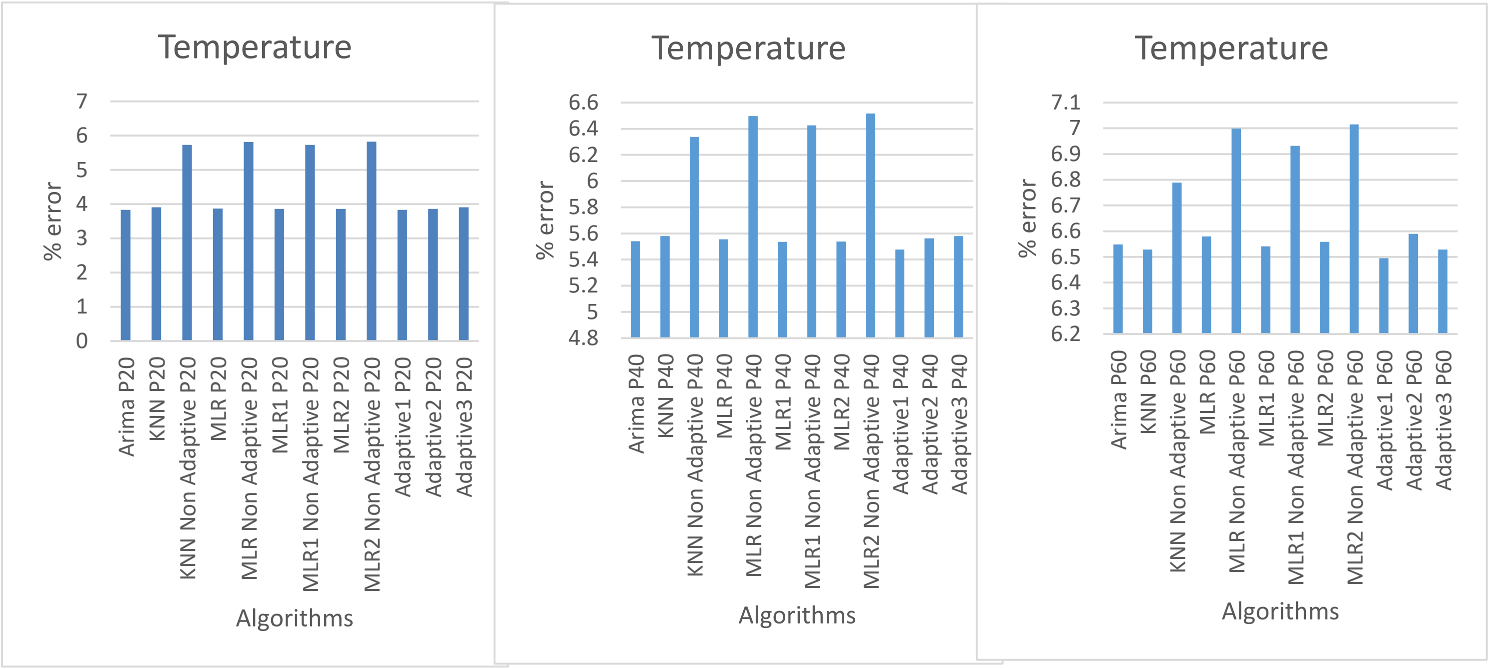
N represents the total number of values recorded and predicted.
The overall average % error O was also computed for all the 7 parameters as follows:

$$O = \frac{1}{7}\sum_{y=1}^{7}E_y \tag{2}$$

The Percentage error for Temperature predicted by all schemes at intervals of 20 mins (P20), 40mins P(40) and 60 mins (P60) and are shown in Figure 5.26.



(a) P20, P40 and P60 shown jointly.



(b) P20 (c) P40 (d) P60

Figure 5.26: Percentage error for Temperature.

The general observation is that as the prediction interval increases from 20 to 60 mins, the % error increases from below 4% to above 6%. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. ARIMA also exhibits a low % error because it has an implicit adaptive function in the sense that it updates its parameters at each new registered value. Figure 5.27 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 5.27% is achieved by the Adaptive 1 scheme. The worst average is obtained by non-adaptive MLR 2 at 6.44%.

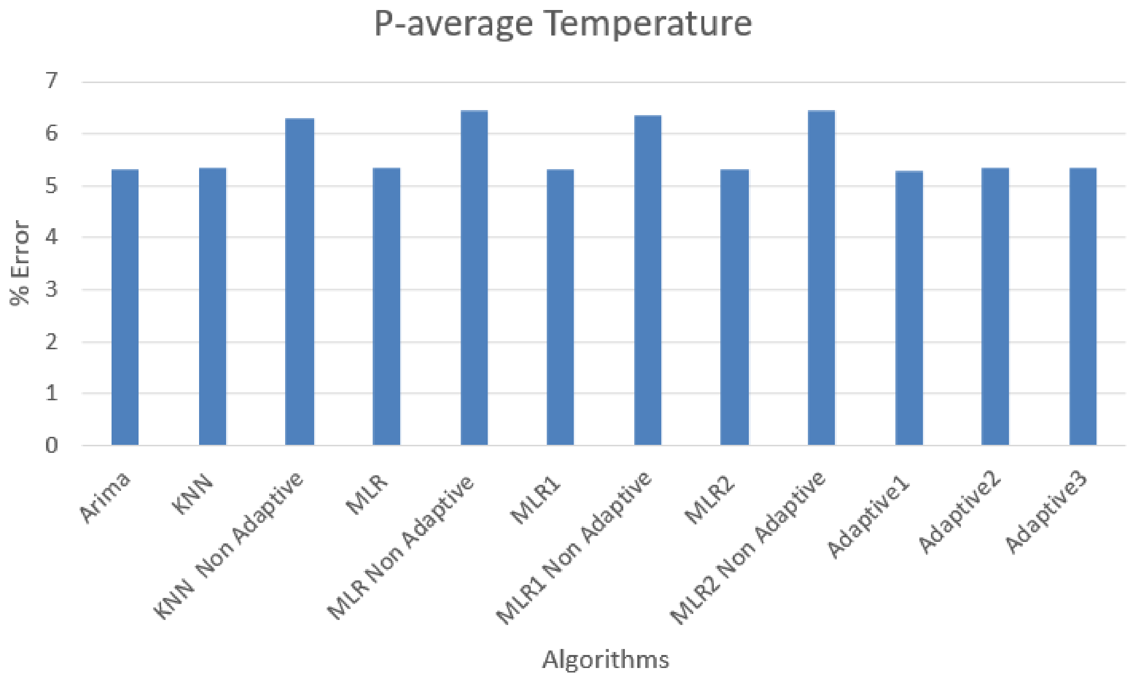
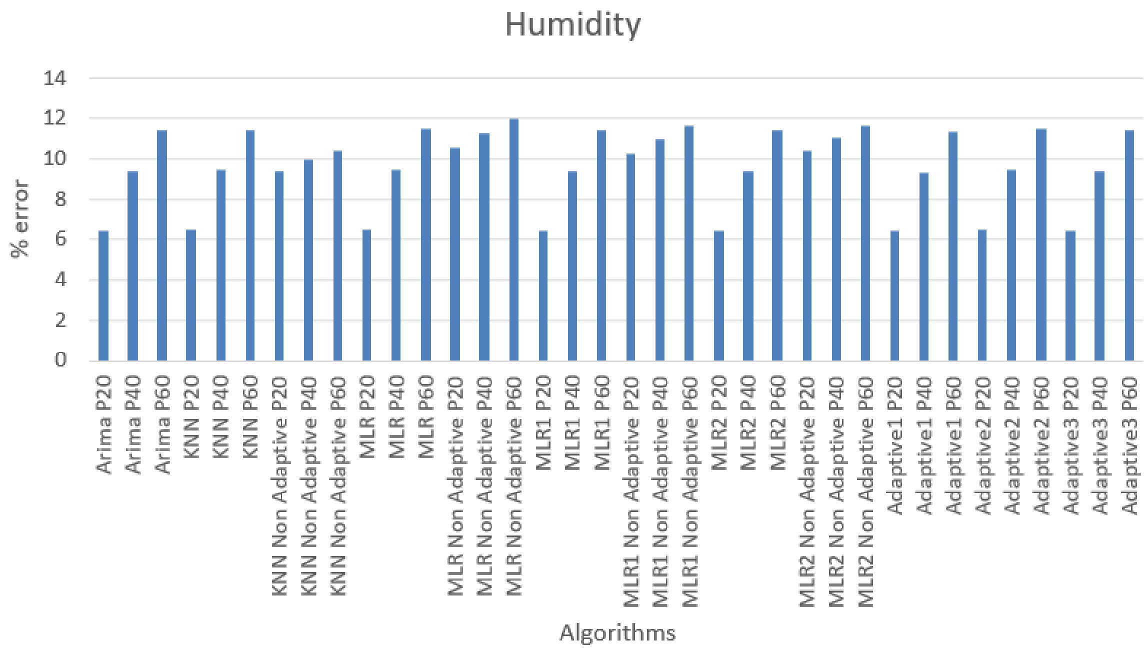
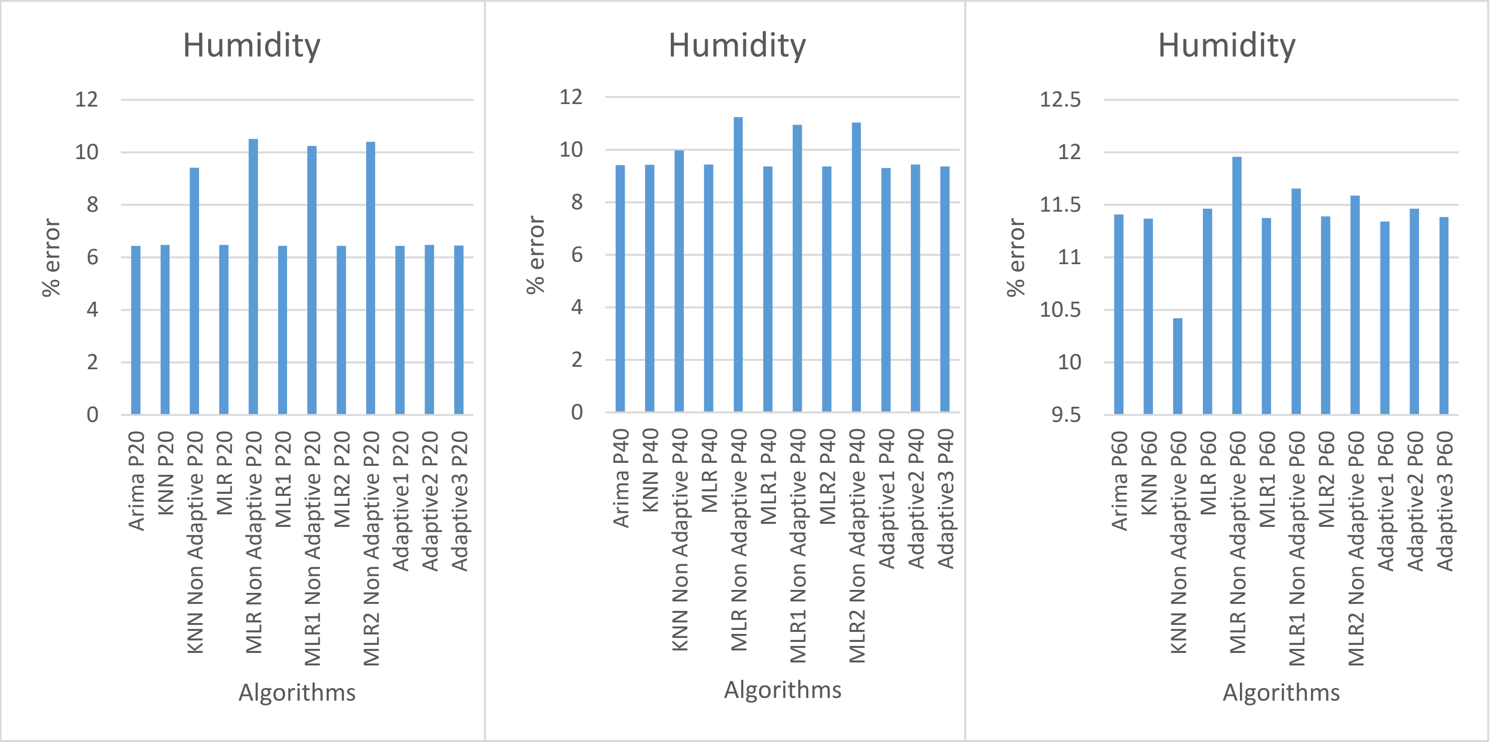


Figure 5.27: P-average for Temperature

The Percentage error for Humidity predicted by all schemes at intervals of 20 mins (P20), 40mins P(40) and 60 mins (P60) and are shown in Figure 5.28.



(a) P20, P40 and P60 shown jointly.



(b) P20

(c) P40

(d) P60

Figure 5.28: Percentage error for Humidity

The general observation is that as the prediction interval increases from 20 to 60 mins, the % error increases from 6.4% to 11.4% for the case of ARIMA for example. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Figure 5.29 shows 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%.

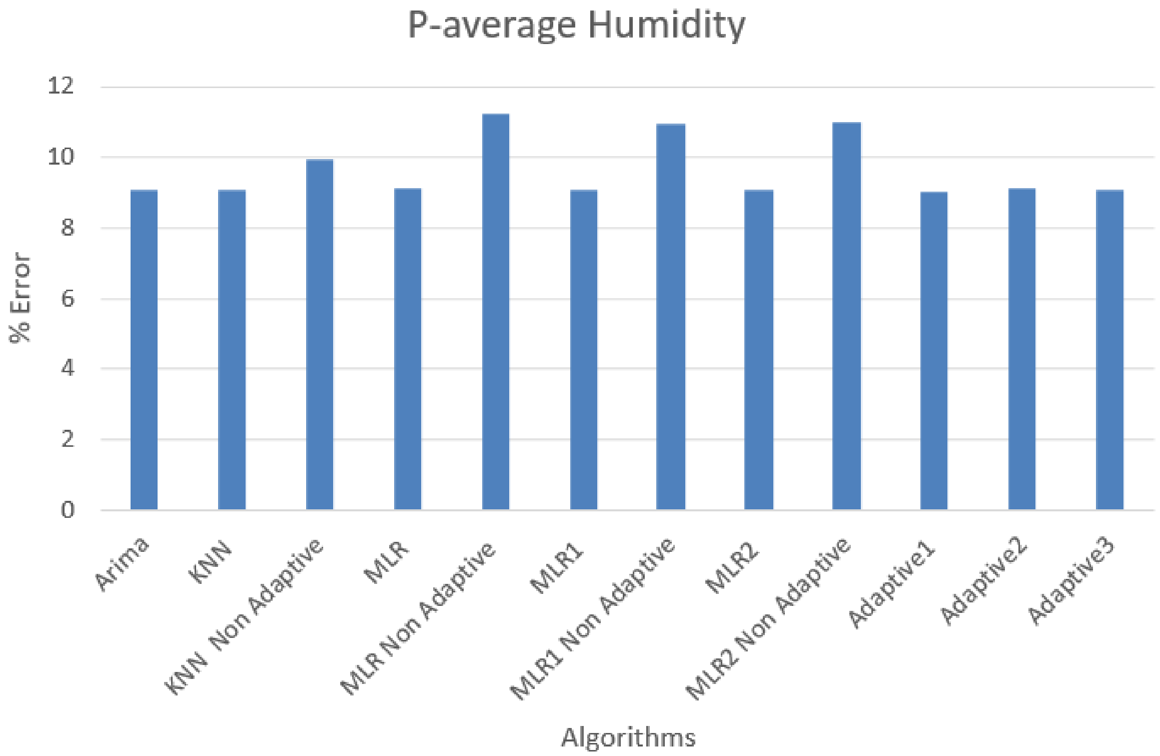
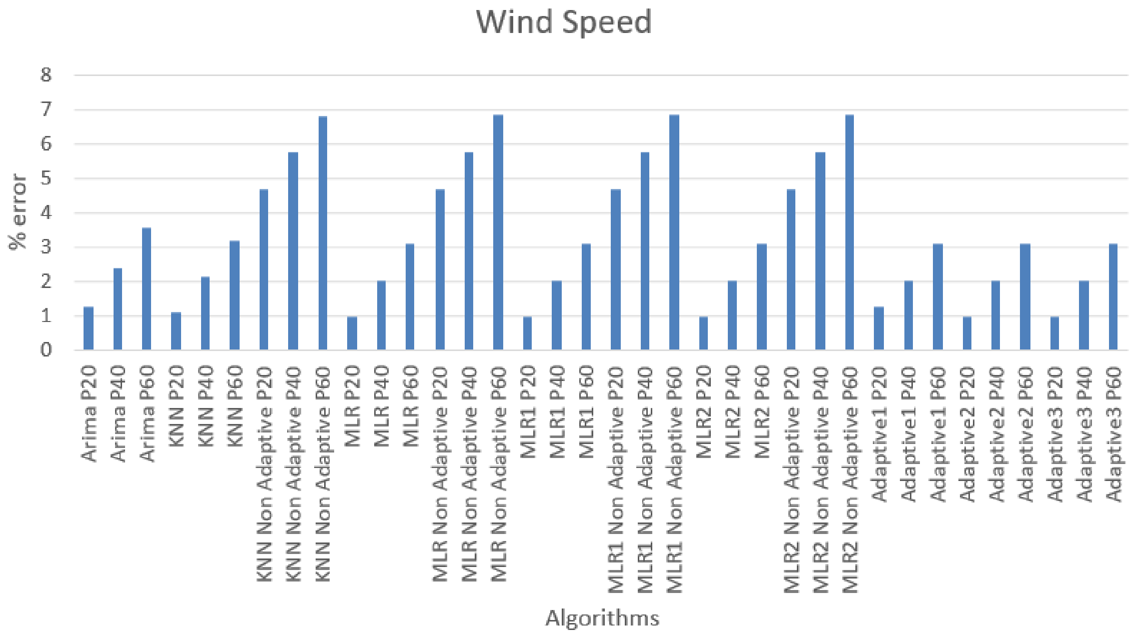
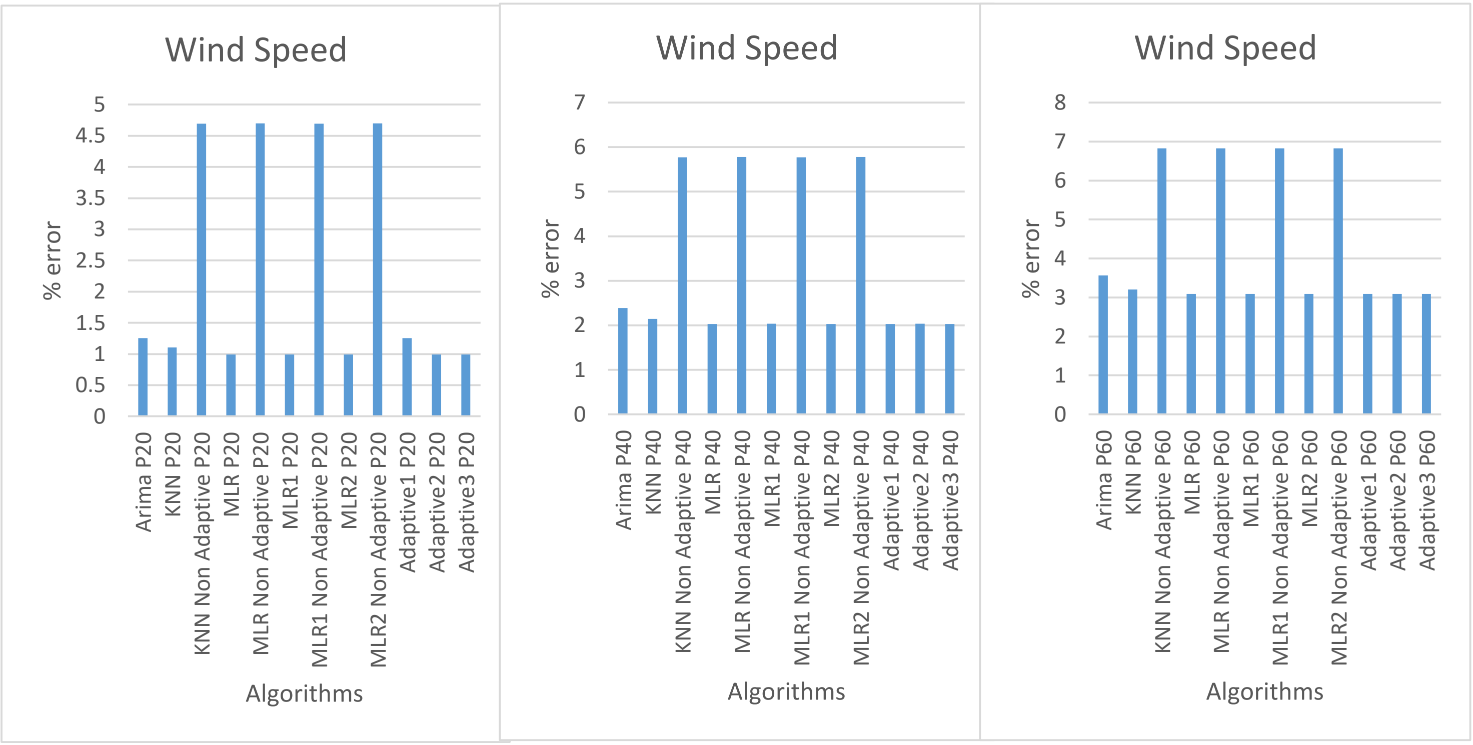


Figure 5.29: P-average for Humidity

The Percentage error for Wind Speed predicted by all schemes at intervals of 20 mins (P20), 40mins P(40) and 60 mins (P60) and are shown in Figure 5.30.



(a) P20, P40 and P60 shown jointly.



(b) P20

(c) P40

(d) P60

Figure 5.31: Percentage error for Wind Speed

The general observation is that as the prediction interval increases from 20 to 60 mins, the % error increases from 0.99% to 3.09% in the case of Adaptive MLR for example. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Figure 5.32 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 2.04% is achieved by the Adaptive 2 scheme. The worst average is obtained by non-adaptive MLR and MLR2 at 5.77%.

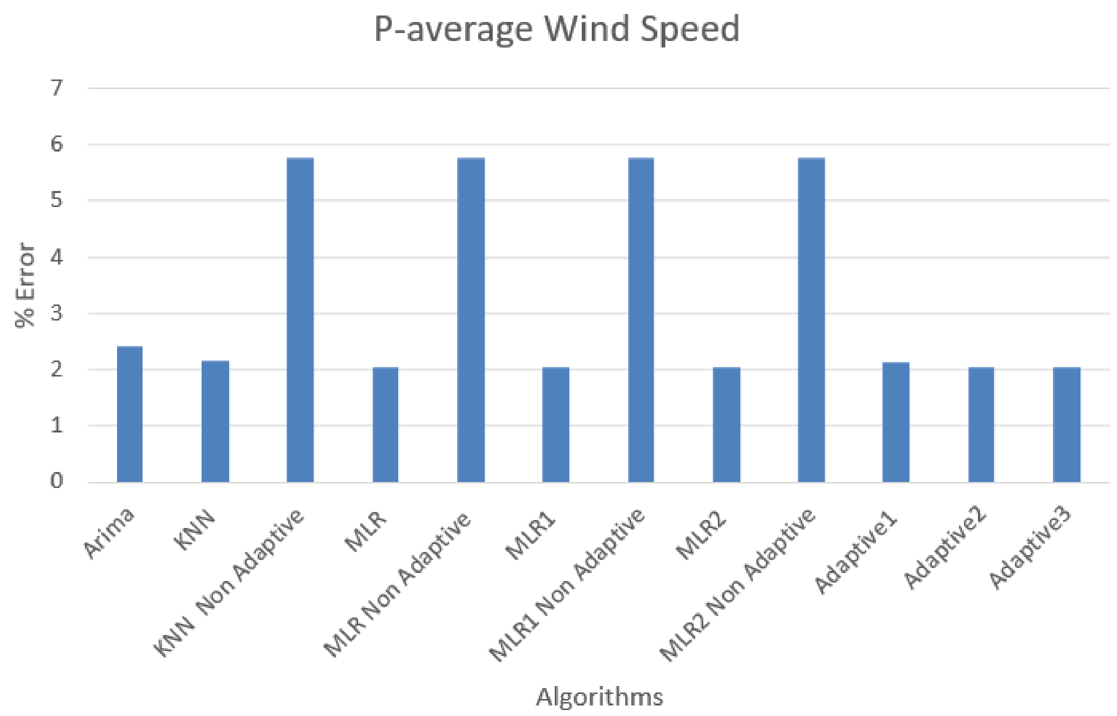
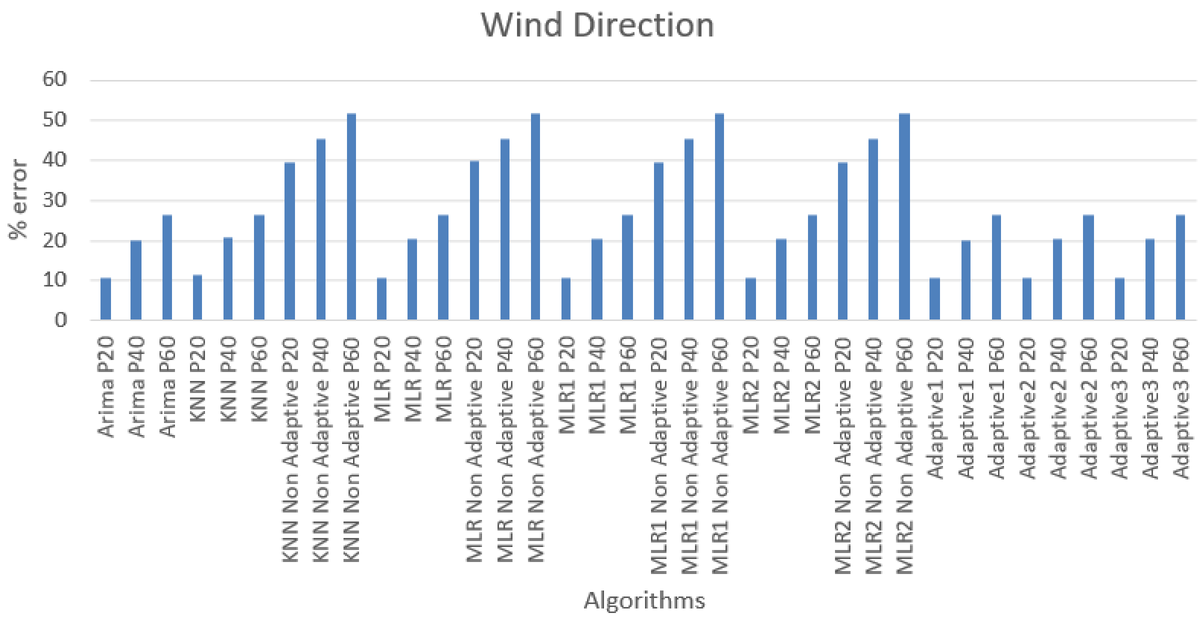
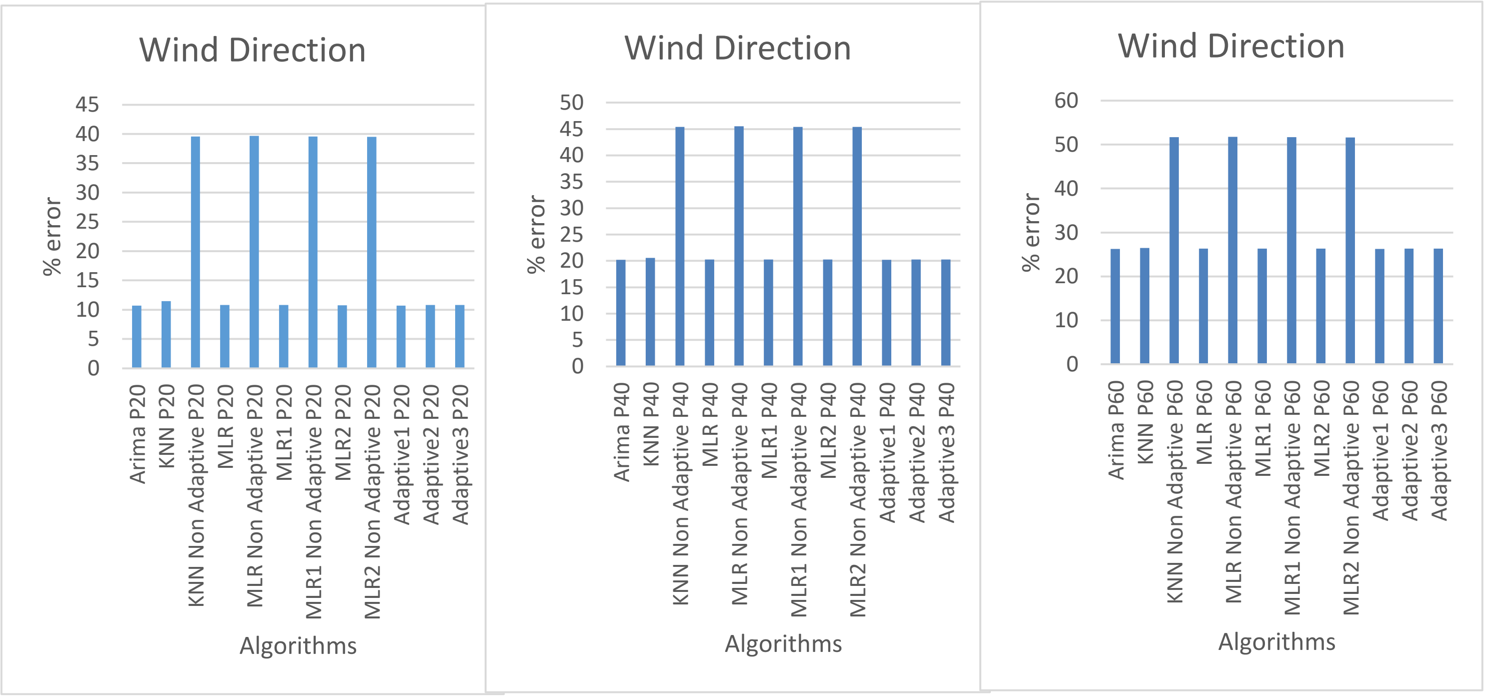


Figure 5.32: P-average for Wind Speed

The Percentage error for Wind Direction predicted by all schemes at intervals of 20 mins (P20), 40mins P(40) and 60 mins (P60) and are shown in Figure 5.33.



(a) P20, P40 and P60 shown jointly.



(b) P20

(c) P40

(d) P60

Figure 5.33: Percentage error for Wind Direction

The general observation is that as the prediction interval increases from 20 to 60 mins, the % error increases from 11.46% to 26.51% in the case of Adaptive KNN for example. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Figure 5.34 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%.

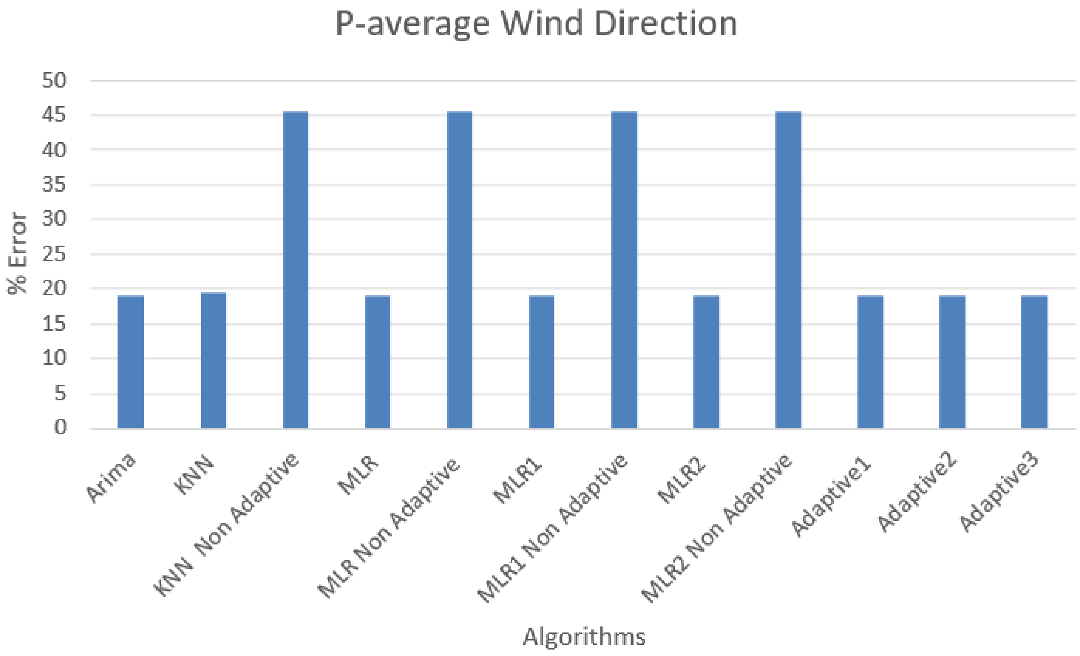
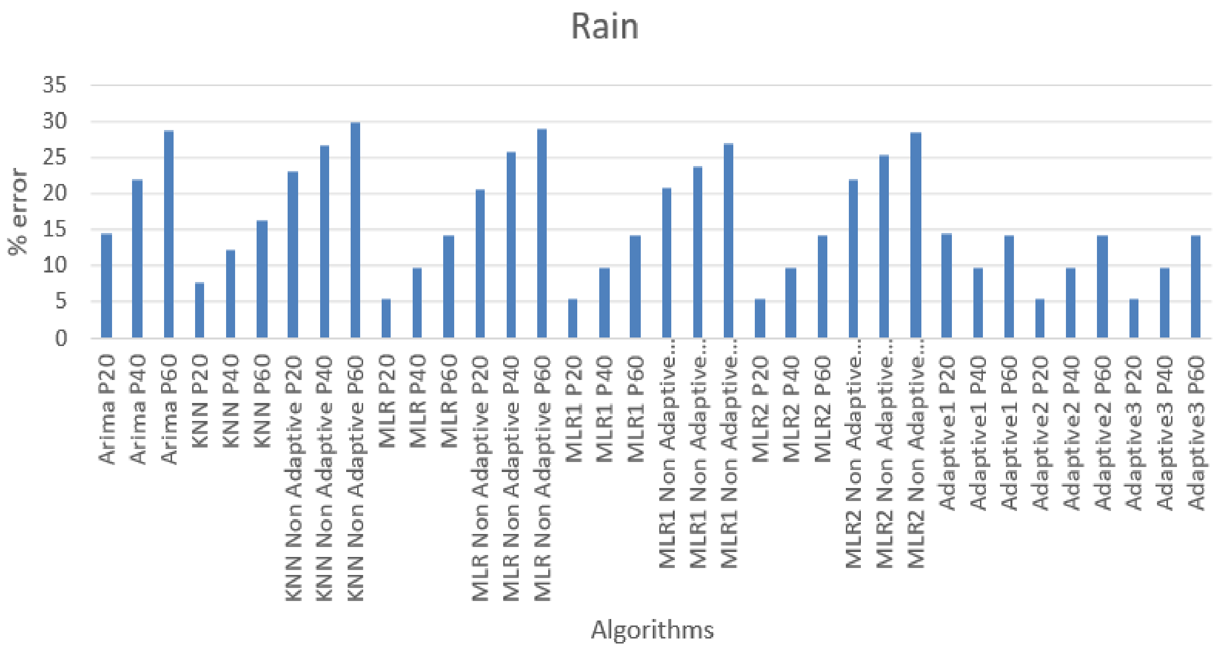
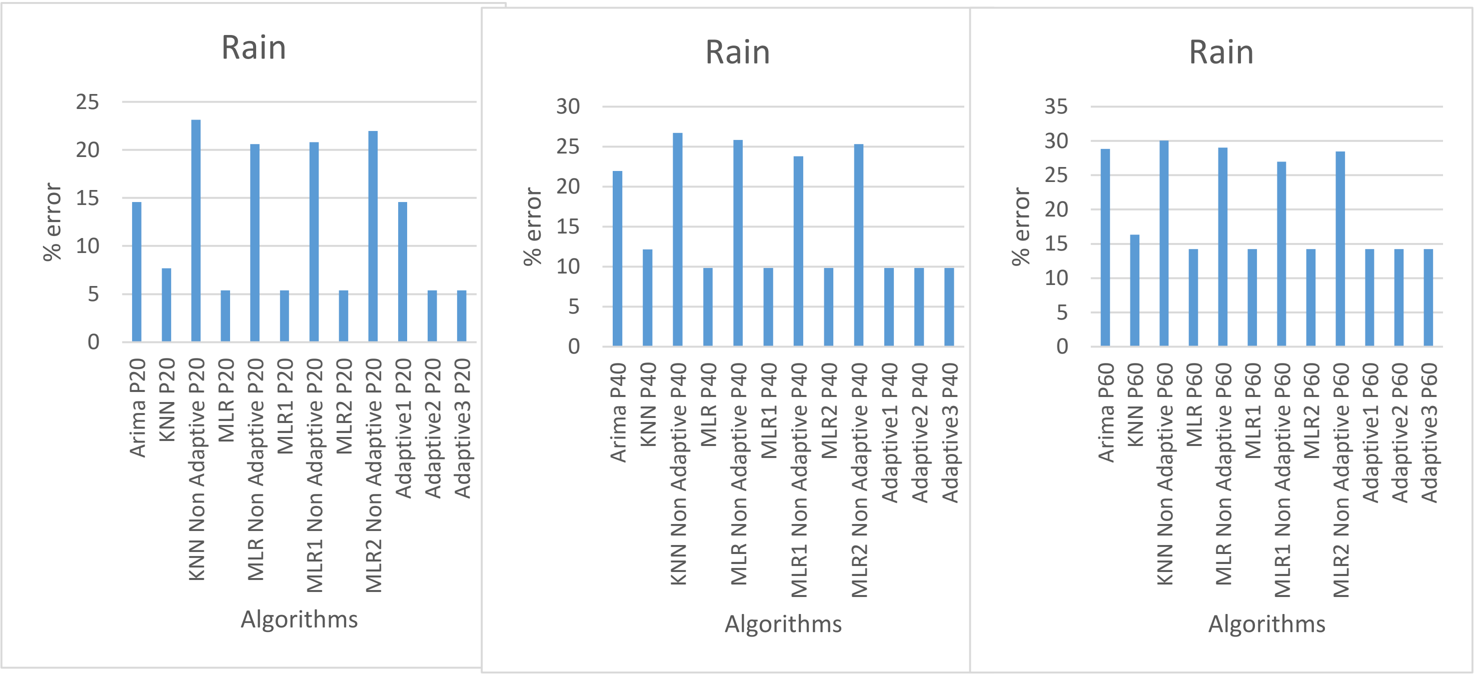


Figure 5.34: P-average for Wind Direction

The Percentage error for Rainfall predicted by all schemes at intervals of 20 mins (P20), 40mins P(40) and 60 mins (P60) and are shown in Figure 5.35. The general observation is that as the prediction interval increases from 20 to 60 mins, the % error increases from 5.39% to 14.22% in the case of Adaptive MLR for example. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones.



(a) P20, P40 and P60 shown jointly.



(b) P20 (c) P40 (d) P60

Figure 5.35: Percentage error for Rain

Figure 5.36 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 9.82% is achieved by the Adaptive MLR 1 scheme. The worst average is obtained by non-adaptive KNN at 26.63%.

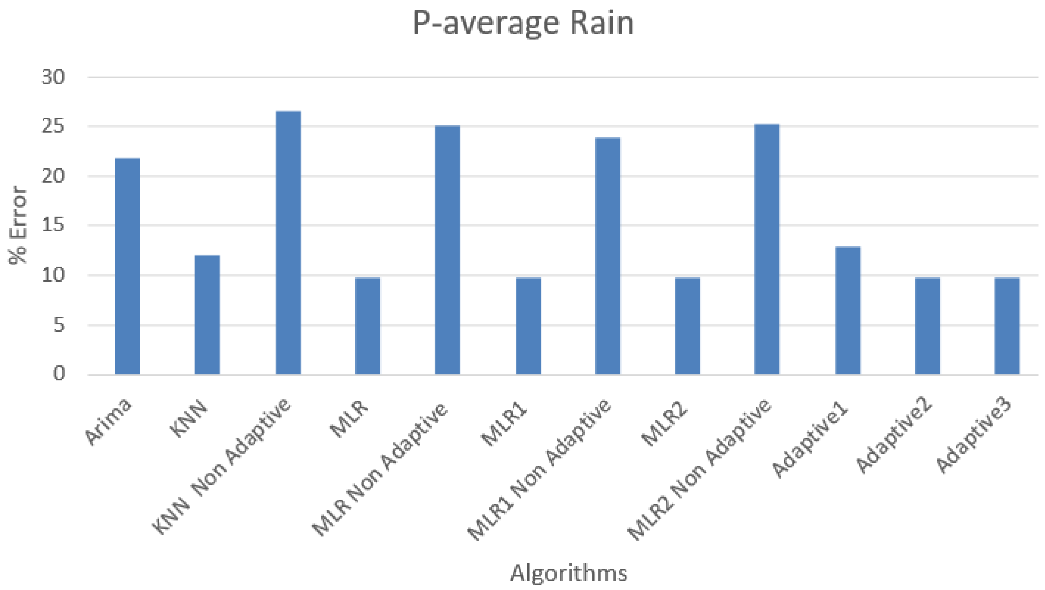
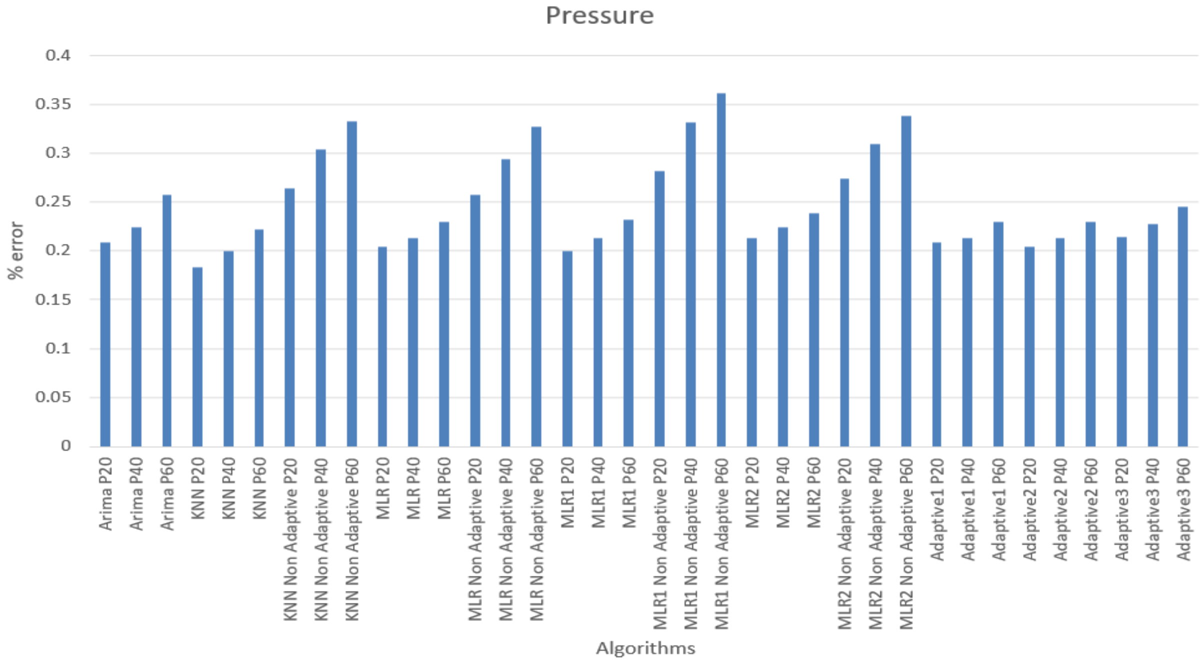
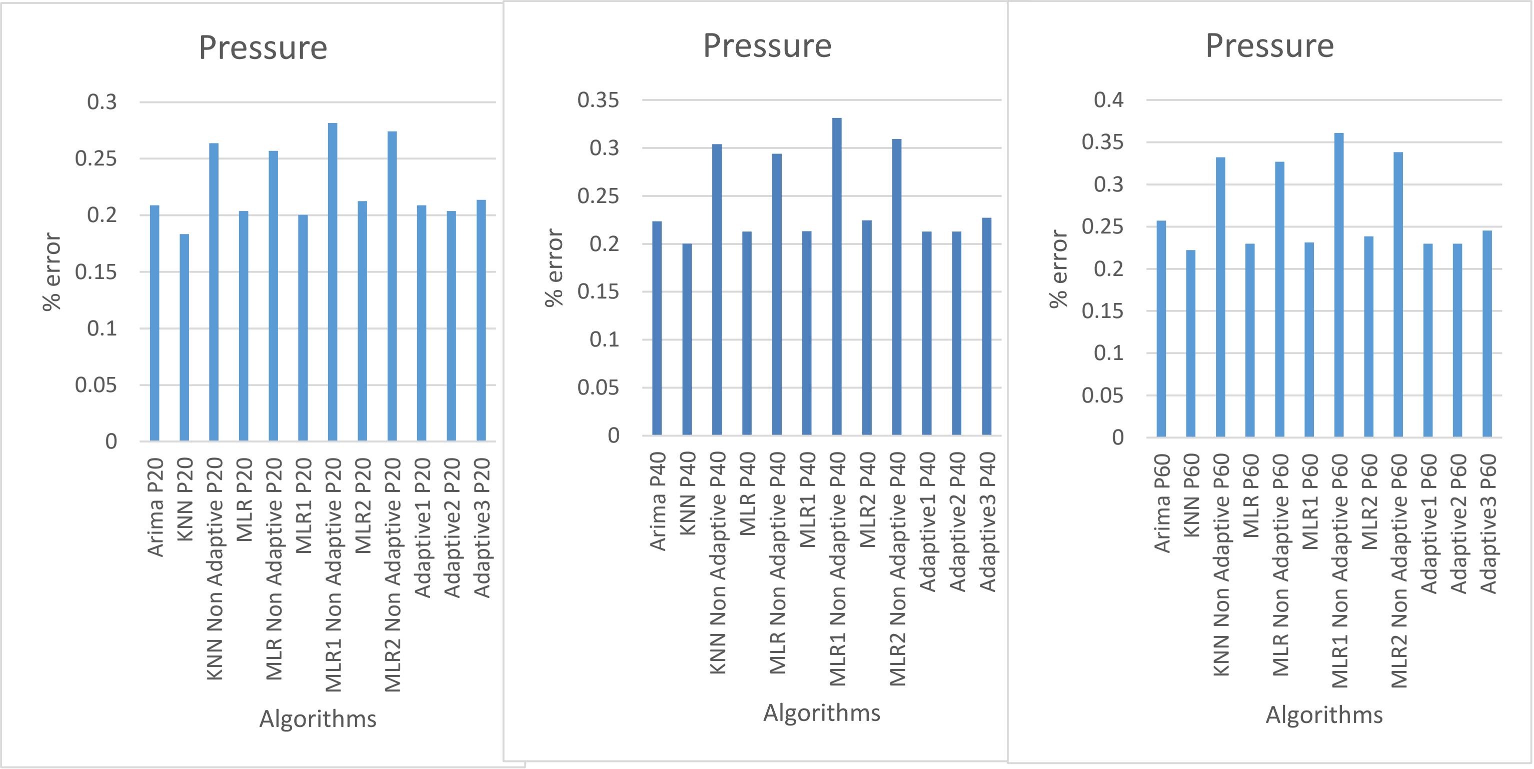


Figure 5.36: P-average for Rain

The Percentage error for Pressure predicted by all schemes at intervals of 20 mins (P20), 40mins P(40) and 60 mins (P60) and are shown in Figure 5.37. The general observation is that as the prediction interval increases from 20 to 60 mins, 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.



(a) P20, P40 and P60 shown jointly.



(b) P20 (c) P40 (d) P60

Figure 5.38: Percentage error for Pressure

Figure 5.39 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 0.215% is achieved by the Adaptive 2 scheme. The worst average is obtained by the non-adaptive MLR 1 scheme at 0.325%.

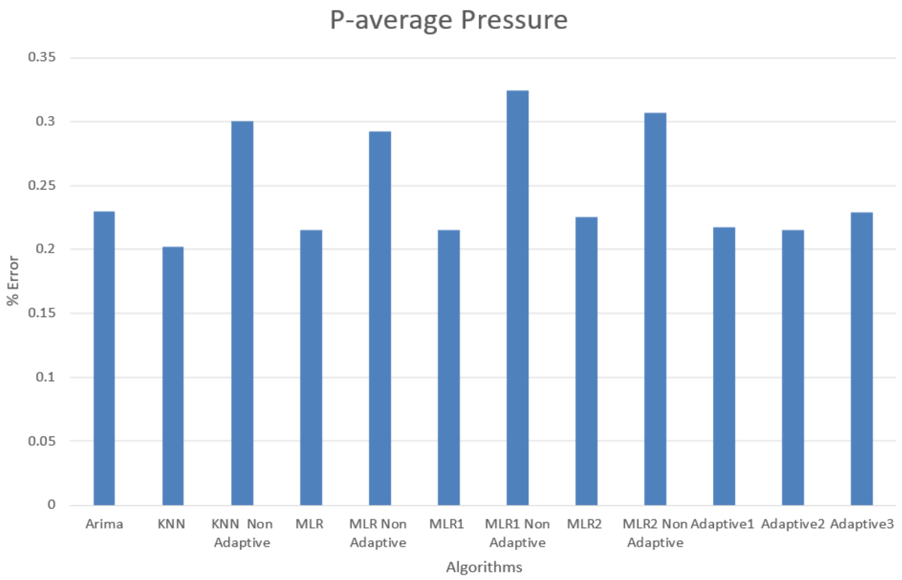
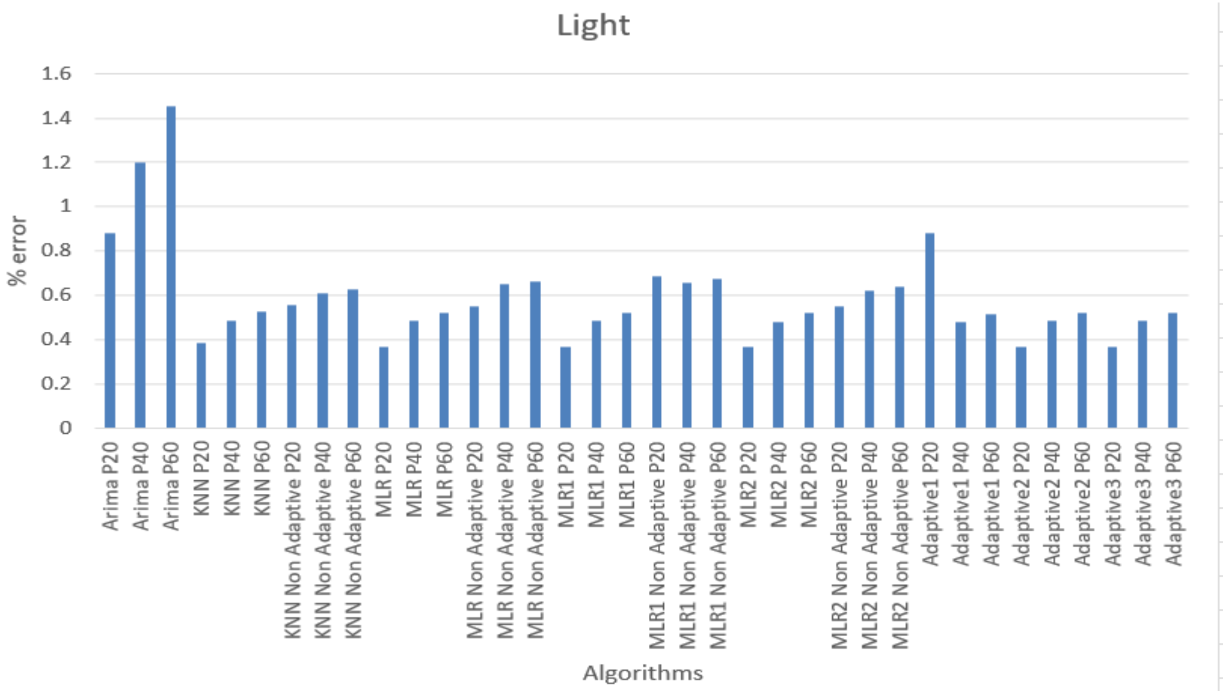
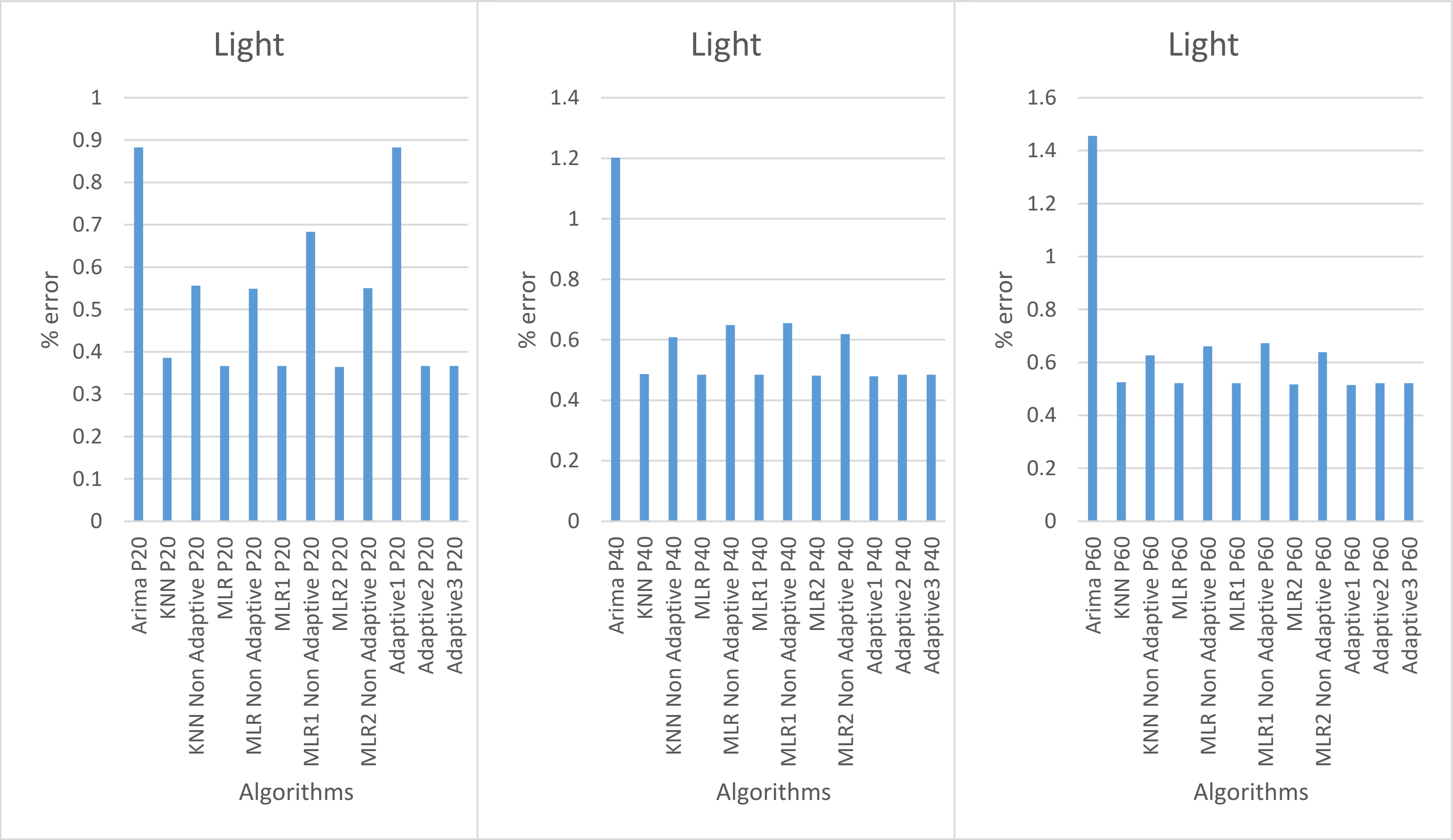


Figure 5.39: P-average for Pressure

The Percentage error for light intensity predicted by all schemes at intervals of 20 mins (P20), 40mins P(40) and 60 mins (P60) and are shown in Figure 5.40. The general observation is that as the prediction interval increases from 20 to 60 mins, the % error increases from 0.88% to 1.46% in the case of ARIMA for example. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Figure 5.15 shows the average prediction for all three intervals i.e. P20, P40 and P60. It is observed that the lowest average % error of 0.456% is achieved by the Adaptive 2 scheme. The worst average is obtained by the ARIMA scheme at 1.18 %.



(a) P20, P40 and P60 shown jointly.



(b) P20 (c) P40 (d) P60

Figure 5.40: Percentage error for Light

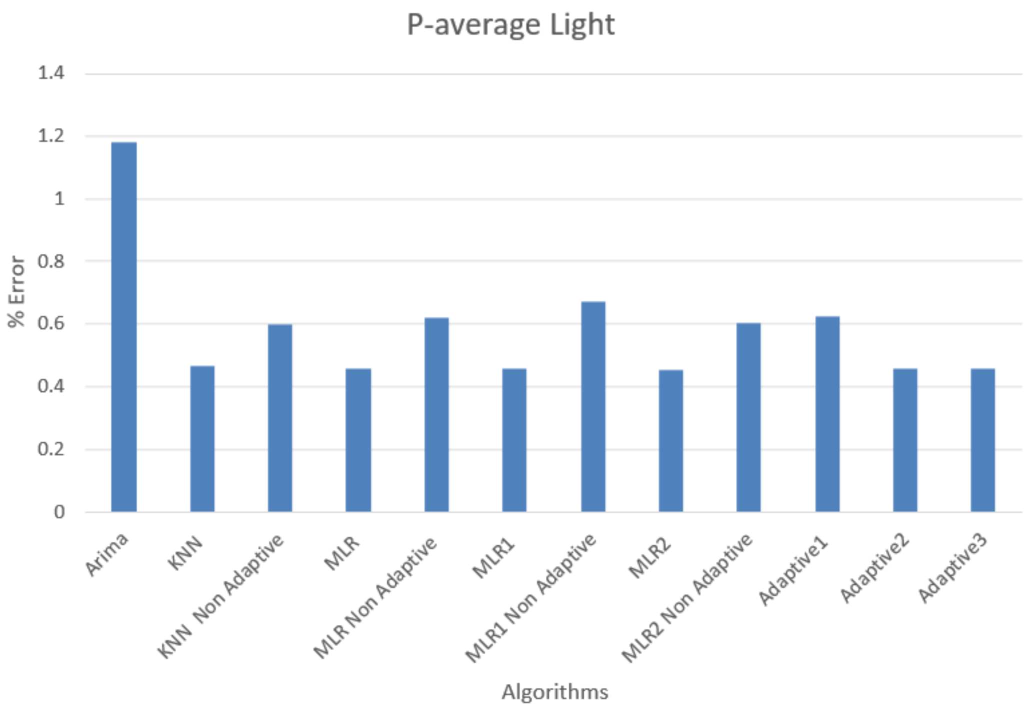


Figure 5.40: P-average for Light

The overall average percentage error for all parameters predicted by all schemes at intervals of 20 mins (P20), 40mins P(40) and 60 mins (P60) are shown in Figure 5.41.

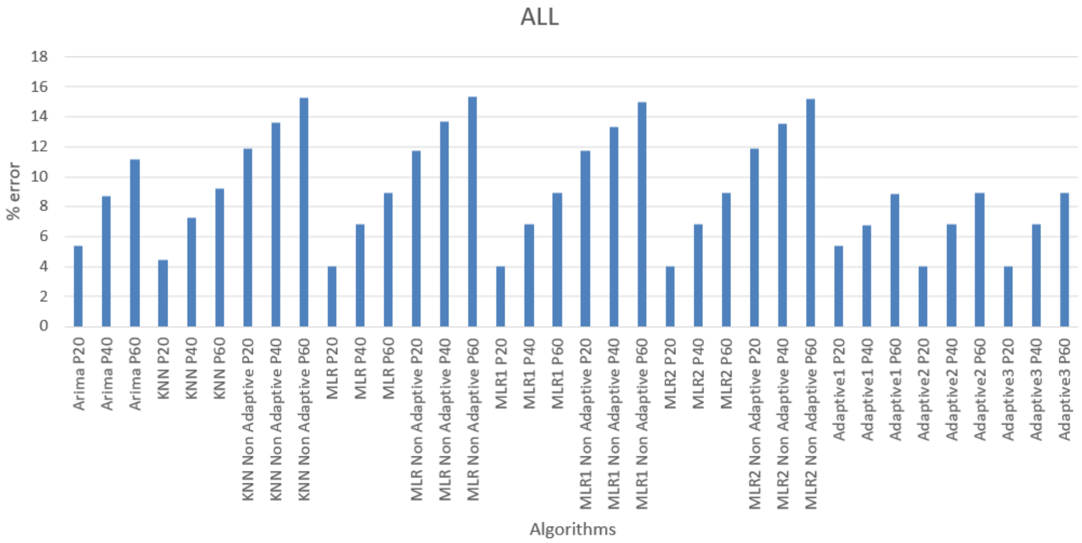


Figure 5.41: Overall percentage error

The general observation is that as the prediction interval increases from 20 to 60 mins, the % error increases from 5.41% to 11.18% in the case of ARIMA for example. Moreover, the % error of all the adaptive schemes is lower than the non-adaptive ones. Figure 5.17 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 %.

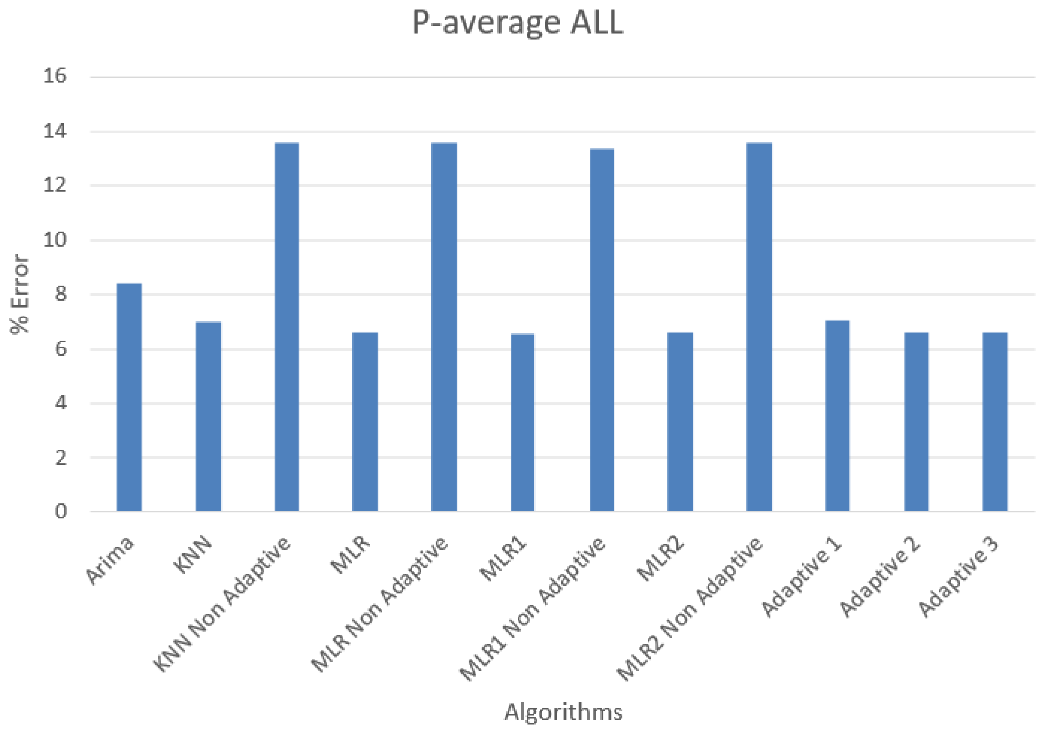


Figure 5.41: Overall P-average

CHAPTER 6
CONCLUSIONS AND FUTURE WORKS

6.1 Conclusions

In this project, three different cloud based real-time weather forecasting systems were developed. The first system consisted of a microcontroller based IoT weather forecasting system which employed a wireless gateway and X-Bee shields for transmitting the recorded sensor values to the control station. The second one was also micro-controller based but used a cabled transmission from the sensor nodes to the control station. The third one employed a dedicated IoT device, the cabled Davis Vantage Pro 2 plus for capturing the weather parameters. All three systems were configured so that they can send all their data on the IBM Bluemix cloud. The programs for performing predictions were hosted and run on the IBM Bluemix platform and benefitted from the High Performance Computing infrastructure of Bluemix. A mobile Application was also developed to allow users to send weather prediction requests to the forecasting application that was hosted on the Bluemix platform and receive the predictions.

Moreover, the performance of the weather forecasting application was 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. Tests were conducted in three different regions, namely the UoM campus at Reduit, Terre Rouge and Vacoas. 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. The short term forecasting can be very useful during a sport tournament or any social event which relies a lot on weather conditions.

Moreover, the experiment conducted in Vacoas for the wind measured under a covered terrace may be useful in the design of a big shopping mall.

6.2 Future works

The system developed in this work can first of all be extended to provide island-wide coverage by installing such weather forecasting units at various points over Mauritius. A typical scenario would be to install the devices in the different districts of Mauritius as depicted in Figure 6.1 [68, 69]. The number of devices per district will depend on the size and geography of the district.

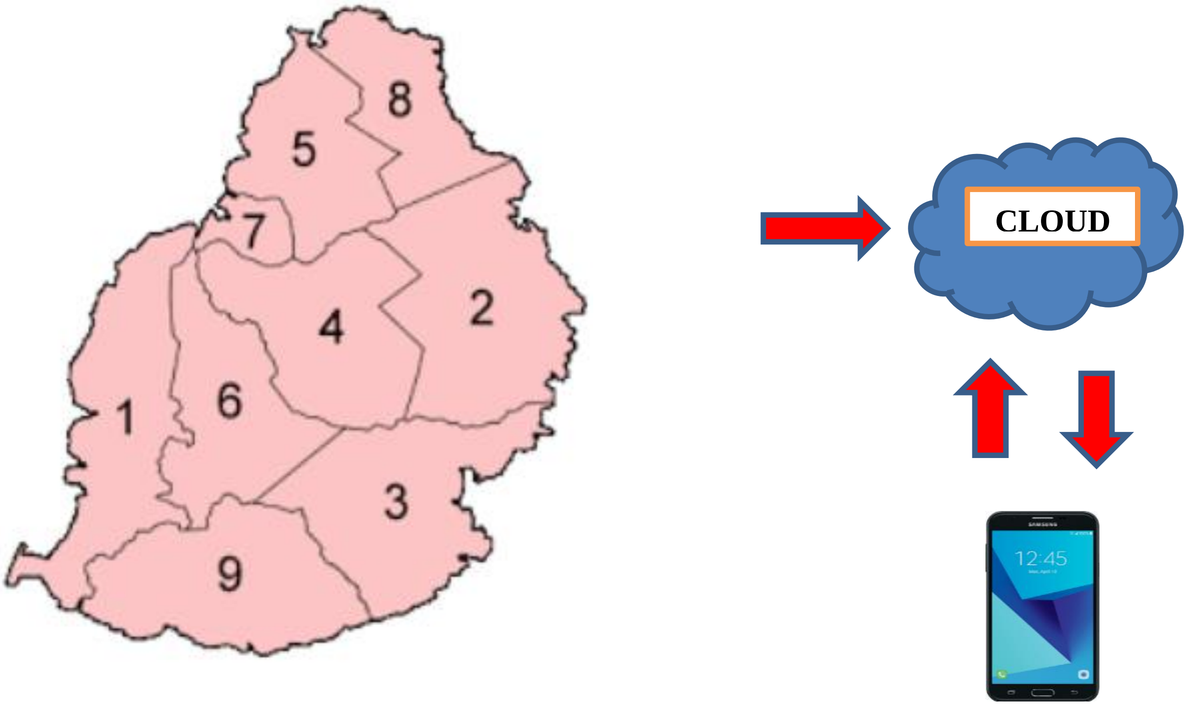


Figure 6.1: Tentative placement plan for IoT weather sensors.

All the data collected by the weather stations will be relayed in real-time to the cloud. As an end-user can connect to the cloud via a mobile application and obtain the state of the current weather conditions as well as short-term forecasts for any of the 23 regions where the system has been installed.

The forecasting model can also be enhanced so that while forecasting a given parameter for a given region, parameters from other regions are also included into the model. This will in fact form a collaborative weather forecasting network as shown in figure 6.2.

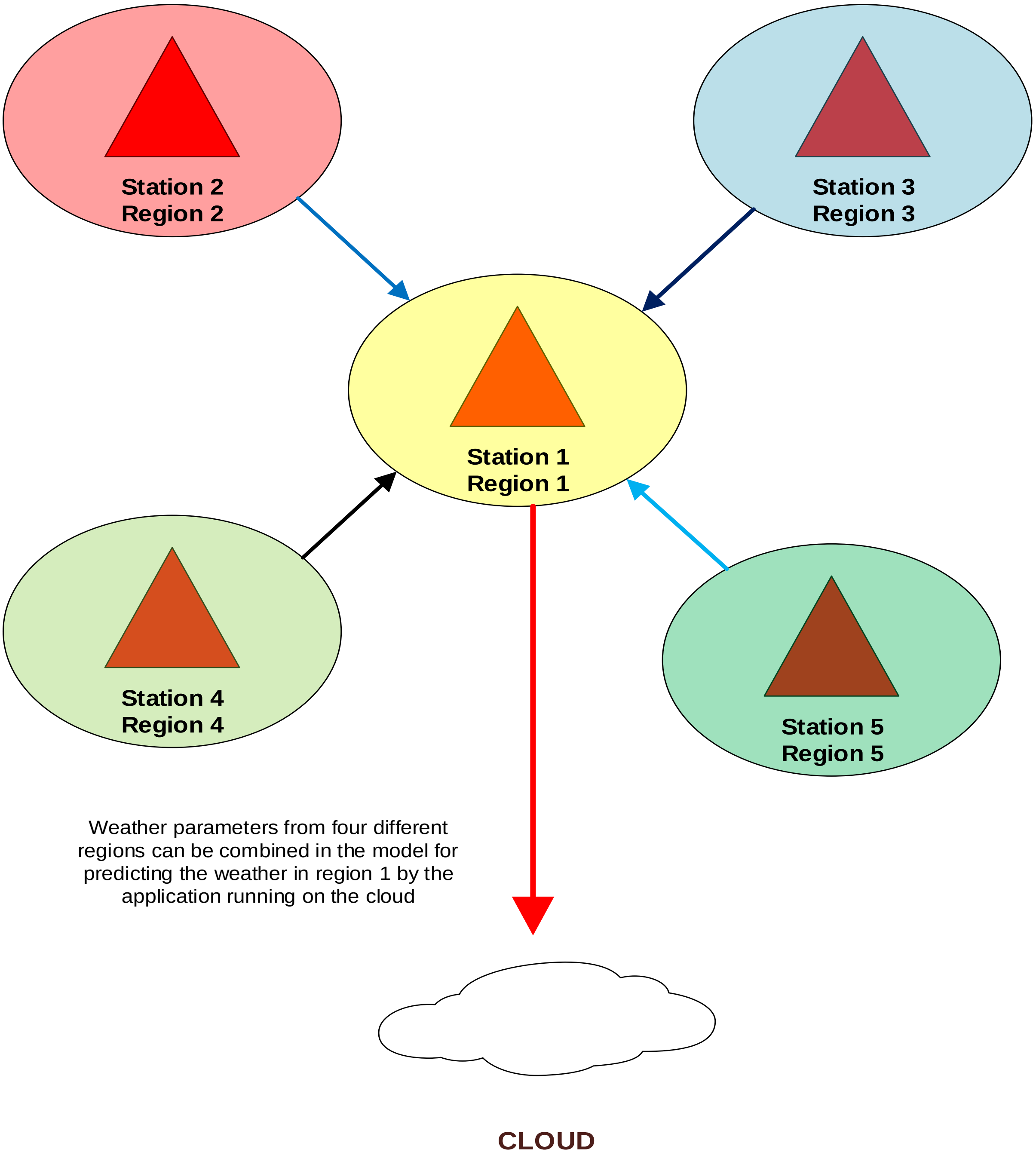

Station 2
Region 2
Station 3
Region 3
Station 1
Region 1
Station 4
Region 4
Station 5
Region 5
Weather parameters from four different regions can be combined in the model for predicting the weather in region 1 by the application running on the cloud
CLOUD

Figure 6.2: Proposed Collaborative Weather Forecasting model.

Finally, the physical basis of the interaction between the different meteorological parameters could be further investigated understood before incorporating them in a regression model equation because a positive/ negative correlation in statistics does not necessarily means that two variables are correlated/uncorrelated. This can lead to fewer equations that will improve the model performance by lowering the error rate and complexity.

References

[1] David M. Karl, Lucas Beversdorf, Karin M. Björkman, Matthew J. Church, Asuncion Martinez & Edward F. Delong, "Aerobic production of methane in the sea" , Nature Geoscience volume 1, pages 473–478 (2008).

[2] National Climatic Data Center, "Climates of the States, Climatography of the United States", No. 60, 2011, Available online: Available at: http://cdo.ncdc.noaa.gov/cgibin/climatenormals/climatenormals.pl

[3] BBC News, 1 January 2011. [Online]. Available: http://www.bbc.com/news/world-asia-pacific-12102126. [Accessed 2016 December 2016].

[4] T. P. Fowdur, Y. Beeharry, V. Hurbungs, V. Bassoo, and V. Ramnarain-Seetohul, "A Framework for a Real-Time Cloud-Based Weather Forecasting System for Mauritius", IJMEC, Vol. 7(26), Oct. 2017, PP. 3563-3581.

[5] G. E. P. Box and G. M. Jenkins, Time Series Analysis: Forecasting and Control. Revised Edition, San Francisco, CA: Holden-Day, 1976.

[6] D. Machiwal and M. K. Jha, "Time Series Analysis of Hydrologic Data for Water ResourcesPlanning and Management: A Review," Journal of Hydrology and Hydromechanics, vol. 54, no.3, pp. 237 – 257, 2006.

[7] Acurite, "5 – in – 1 Weather Sensor," 2010 – 2015. [Online]. Available: http://www.acurite.com/learn/weather-stations/acurite-5-in-1-sensor. [Accessed 11 June 2016].

[8] Arpit Tiwari, S K Verma, "Cloudburst Predetermination System", IOSR Journal of Computer Engineering (IOSR-JCE)Volume 17, Issue 3, Ver. V (May –Jun. 2015), PP 47-56

[9] B. M. Padya, Weather and Climate of Mauritius (1st Edition), Moka: Mahatma Gandhi Institute, Mauritius, 1989.

[10] S. C. Fowdur, S. D. D. V. Rughooputh, J. Cheeneehash, R. Boojhawon and A. Gopaul, "Rainfall analysis over Mauritius using Principal Component Analysis," Environmental Management and Sustainable Development, vol. 3, no. 2, pp. 94 – 108, 2014.

[11] J. M. Chun, "Migration, Environment and Climate Change: Policy Brief Series," International Organisation for Migration, vol. 1, no. 11, 2015.

[12] Mauritius Meteorological Services, "Climate Change," 2016. [Online]. Available: metservice.intnet.mu/climate-services/climate-change.php. [Accessed 16 December 2016].

[13] Meteorological Services, "Technical Report CS 28 Cyclone Season of the South West Indian Ocean 2006-2007," January 2008. [Online]. Available: http://metservice.intnet.mu/pdfs/technical-note-on-cyclone-season.pdf. [Accessed 14 December 2016].

[14] Ministry of Environment and Sustainable Development, "Maurice Ile Durable: Policy, Strategyand Action Plan," May 2013. [Online]. Available:http://www.govmu.org/portal/sites/mid/file/full%20report%20midpolicy.pdf. [Accessed 13December 2016].

[15] Intergovernmental Panel on Climate Change, "Impacts, Adaptation and Vulnerability.Contribution of Working Group II to the Fourth Assessment Report of the IntergovernmentalPanel on Climate Change," Cambridge University Press, Lipton, 2007.

[16] Government Information Service Newsletter, "Eleven die in flash floods," March 2013.[Online]. Available: http://gis.govmu.org/English/Documents/News%20March%202013.pdf. [Accessed 14 December 2016].

[17] Ministry of Environment and Sustainable Development, and Disaster and Beach Management,"National Climate Change Adaptation Policy Framework for the Republic of Mauritius," 30 July2013. [Online]. Available:http://environment.govmu.org/English/Climate_Change/Pages/Climate-Change.aspx.[Accessed 14 December 2016].

[18] MAURITIUS METEOROLOGICAL SERVICES, CLIMATE JANUARY 2019 [Available Online]:
http://metservice.intnet.mu/mmsimages/2019-02-27_15:46:36_Monthly%20Climate%20%20Bulletin%20for%20January%202019.pdf

[19] Bulipe Srinivas Rao, K. Srinivasa Rao, and N. Ome, "Internet of Things (IOT) Based Weather Monitoring system," International Journal of Advanced Research in Computer and Communication Engineering, vol. 5, no. 9, pp. 312-319, 2016.

[20] PRACHI H. KULKARNI and PRATIK D. KUTE, "INTERNET OF THINGS BASED SYSTEM FOR REMOTE MONITORING OF WEATHER PARAMETERS AND APPLICATIONS," International Journal of Advances in Electronics and Computer Science, vol. 3, no. 2, pp. 68-73, 2016.

[21] Karthik Krishnamurthi, Suraj Thapa, Lokesh Kothari, and Arun Prakash, "Arduino Based Weather Monitoring System," International Journal of Engineering Research and General Science, vol. 3, no. 2, pp. 452-458, 2015.

[22] Adil Hamid Malik, Aaqib jalal, and Bilal Ahmed Parray, "Smart City IoT Based Weather Monitoring System," International Journal of Engineering Science and Computing, vol. 7, no. 6, pp. 12123-12128, 2017.

[23] Aneeta Varghese, "Weather Based Information System using IoT and Cloud Computing," Journal of Computer Science and Engineering, vol. 2, no. 6, pp. 90-97, 2015.

[24] Arpit Tiwari and S K Verma, “Cloudburst Predetermination System,” Journal of Computer Engineering, vol. 17, no. 3, pp. 47-56, 2015.

[25] Joel T. de Castro, Gabriel M. Salistre, Jr, Young-Cheol Byun, and Bobby D. Gerardo, “Flash Flood Prediction Model based on Multiple Regression Analysis for Decision Support System,” , San Francisco, USA, 2013.

[26] Massimo Ancona, Andrea Dellacasa, Giorgio Delzanno, Andrea La Camera, and Ivano Rellini, “An “Internet of Things” Vision of the Flood Monitoring Problem,” , Nice, France, 2015.

[27] Dilip Kumar Krishnappa, David Irwin, Eric Lyons, and Michael Zink, “CloudCast: Cloud computing for short-term mobile weather forecasts,” , Austin, TX, USA, 2012.

[28] Sheikh Azid et al., “SMS BASED FLOOD MONITORING AND EARLY WARNING SYSTEM,” Journal of Engineering and Applied Sciences, vol. 10, no. 15, pp. 6387-6391, 2015.

[29] Paras and Sanjay Mathur, “A Simple Weather Forecasting Model Using Mathematical Regression,” Indian Research Journal of Extension Education, vol. 1, no. Special Issue, p. 161, 2012.

[30] Hossein Ansari, “Forecasting Seasonal and Annual Rainfall Based on Nonlinear Modeling with Gamma Test in North of Iran,” International Journal of Engineering Practical Research, vol. 2, no. 1, p. 16, 2013.

[31] Retius Chifurira and Delson Chikobvu, “A Weighted Multiple Regression Model to Predict Rainfall Patterns: Principal Component Analysis approach,” Mediterranean Journal of Social Sciences, vol. 5, no. 7, p. 34, 2014.

[32] S. Prabakaran, P. Naveen Kumar , and P. Sai Mani Tarun, “Rainfall prediction using modified linear regression,” ARPN Journal of Engineering and Applied Sciences, vol. 12, no. 12, p. 3715, 2017.

[33] Leixiao Li, Zhiqiang Ma, Limin Liu, and Yuhong Fan, “Hadoop-based ARIMA Algorithm and its Application in Weather Forecast,” International Journal of Database Theory and Application, vol. 6, no. 5, p. 119, 2013.

[34] Ernesta GRIGONYTĖ and Eglė BUTKEVIČIŪTĖ, “Short-term wind speed forecasting using ARIMA model,” ENERGETIKA, vol. 62, no. 1-2, p. 45, 2016.

[35] Seyed Amir Shamsnia, Naeem Shahidi, Ali Liaghat, Amirpouya Sarraf, and Seyed Farnood Vahdat, “Modeling of Weather Parameters Using Stochastic Methods (ARIMA Model)(Case Study: Abadeh Region, Iran),” International Conference on Environment and Industrial Innovation, vol. 12, no. 2011, p. 282, 2011.

[36] Inderjeet Kaushik and Sabita Madhvi Singh, “SEASONAL ARIMA MODEL FOR FORECASTING OF MONTHLY RAINFALL AND TEMPERATURE,” Environmental Research And Development, vol. 3, no. 2, p. 506, 2008.

[37] Godwin Monday Datong and Emmanuel Nengak Goltong, “An Application of ARIMA Models in Weather Forecasting: A Case Study of Heipang Airport – Jos Plateau, Nigeria,” Innovative Scientific & Engineering, vol. 5, no. 2, pp. 1-13, 2017.

[38] Sagar S. Badhiye, Nilesh U. Sambhe, and P. N. Chatur, “KNN Technique for Analysis and Prediction of Temperature and Humidity Data,” International Journal of Computer Applications (0975 – 8887), vol. 61, no. 14, p. 7, 2013.

[39] Mohammad Bannayan and Gerrit Hoogenboom, “Weather analogue: A tool for real-time prediction of daily weather data realizations based on a modified k-nearest eighbour approach,” Environmental Modelling & Software, vol. 23, no. 2008, pp. 703-713, 2007.

[40] Zhao Liu and Ziang Zhang, “Solar Forecasting by K-Nearest Neighbors Method with Weather Classification and Physical Model,” , Denver, CO, USA, 2016.

[41] Muhammad Abrar, Anwar Mirza, Zahoor Jan, and Shariq Bashir, “Seasonal to Inter-annual Climate Prediction Using Data Mining KNN Technique,” , Jamshoro, Pakistan, 2008.

[42] Mohammad Bannayan and Gerrit Hoogenboom, “Predicting realizations of daily weather data for climate forecasts using the non-parametric nearest-neighbour re-sampling technique,” INTERNATIONAL JOURNAL OF CLIMATOLOGY, vol. 28, no. 2008, pp. 1357–1368, 2007.

[43] Fhira Nhita, Deni Saepudin, Adiwijaya, and Untari Novia Wisesty, “Comparative Study of Moving Average on Rainfall Time Series Data For Rainfall Forecasting Based on Evolving Neural Network Classifier,” , Bali, Indonesia, 2015.

[44] Kumar Abhishek, M. P. Singh, Saswata Ghosh, and Abhishek Anand, “Weather forecasting model using Artificial Neural Network,” Science direct, vol. 4, no. 2012, pp. 311-318, 2012.

[45] S. Santhosh Baboo and I. Kadar Shereef, “An Efficient Weather Forecasting System using Artificial Neural Network,” International Journal of Environmental Science and Development, vol. 1, no. 4, pp. 321 – 326, 2010.

[46] Gerben Meijer, “Predicting the Dutch Weather Using Recurrent Neural,” Enschede, 2017.

[47] Saktaya Suksri and Warangkhana Kimpan, “Neural Network Training Model for Weather Forecasting Using Fireworks Algorithm,” , Chiang Mai, Thailand, 2016.

[48] Jayrani Cheeneebash, Ashvin Harradon, Ashvin Gopaul, “Forecasting Rainfall in Mauritius using Seasonal Autoregressive Integrated Moving Average and Artificial Neural Networks”, Environmental Management and Sustainable Development, Vol 7, No 1 (2018).

[49] Monidipa Das and Soumya K.Ghosh, “Data-driven approaches for meteorological time series prediction: A comparative study of the state-of-the-art computational intelligence techniques” Pattern Recognition Letters, Volume 105, 1 April 2018, Pages 155-164.

[50] Sanford Weisberg, “Applied Linear Regression”, Wiley Series in Probability and Statistics, 4th Edition, Dec 2013.

[51] OriginLab. OriginLab. [Online]. http://www.originlab.com/doc/Origin-Help/Multi-Regression-Algorithm.

[52] R J Hyndman and G Athanasopoulos, Forecasting: Principles and Practice, 1st ed. Australia: Monash University, 2017.

[53] Richard O. Duda, Peter E. Hart, and David G. Stork, Pattern Classification.: John Wiley & Sons, Inc, 2000.

[54] S. S. Haykin and R. Gwynn, Neural networks and learning machines.: Prentice Hall, 2008.

[55] D E Rumelhart, G E Hinton, and R J Williams, Learning internal representations by error propagation. USA: Parallel distributed processing: explorations in the microstructure of cognition, 1986.

[56] Wikibooks. (2017) Artificial Neural Networks/Neural Network Basics. [Online]. https://en.wikibooks.org/wiki/Artificial_Neural_Networks/Neural_Network_Basics

[57] J. Sola and Joaquin Sevilla, “Importance of input data normalization for the application of neural networks to complex industrial problems,” IEEE, vol. 44, pp. 1464-1468, 1997.

[58] Java Neural Network Framework, Neuroph, http://neuroph.sourceforge.net/

[59] (2017) https://www.arduino.cc/. [Online]. https://www.arduino.cc/en/Guide/Environment

[60] Sparkfun. (2017) https://www.sparkfun.com/. [Online]. https://www.sparkfun.com/products/11021#features-tab

[61] Raspberry Pi. (2017) Raspberrypi.org. [Online]. https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[62] Raspberry Pi. (2017) raspberrypi.org. [Online]. https://www.raspberrypi.org/downloads/raspbian/

[63] Sparkfun. (2017) Sparkfun. [Online]. https://www.sparkfun.com/products/12847

[64] SparkFun. (2017) SparkFun. [Online]. https://www.sparkfun.com/products/13676

[65] Sparkfun. (2017) Sparkfun. [Online]. https://www.sparkfun.com/products/8942

[66] https://www.davisinstruments.com/solution/vantage-pro2/

[67] https://www.ibm.com/cloud/garage/toolchains

[68] P. Goolaup, “Mauritius Meteorological Services – Capabilities for the provision of climate services at national level,” 14 March 2016. [Online]. Available: http://www.gfcs-climate.org/sites/default/files/Mauritius.pdf. [Accessed 14 December 2016].

[69] http://mapsof.net/mauritius/mauritius-districts-numbered

Appendix A1 : T-Test for the significance of results obtained.

It is observed that the predicted values are very close to the actual values. The difference between the mean of the predicted values and that of the actual is in fact insignificant as revealed by the independent samples t-tests performed in the following tables. It is observed that in most cases the absolute value of t-obs is less than the absolute value of t-table for all schemes.

Temperature Day 1 (23rd February 2018)

Algorithm	Mean	Tobserved	Ttable	Significance
Actual	26.2340	---	----	-----
predicted_adaptive_train_win_avg	26.5745	-0.0618	1.6794	NO
predicted_adaptive_train_win_count	26.6099	-0.0682	1.6794	NO
predicted_adaptive120	26.5957	-0.0657	1.6794	NO
predicted_arima120	26.5957	-0.0657	1.6794	NO
predicted_knn120	26.6099	-0.0682	1.6794	NO
predicted_multi_linearregression1_120	26.5851	-0.0638	1.6794	NO
predicted_multi_linearregression2_120	26.5713	-0.0613	1.6794	NO
predicted_multi_linearregression120	26.5762	-0.0622	1.6794	NO

If Tobs > Ttable : Sig = Yes   Expected : No

Humidity Day 1 (23rd February 2018)

Algorithm	Mean	Tobserved	Ttable	Significance
Actual	74.3617	---	----	-----
predicted_adaptive_train_win_avg	73.1718	0.0774	1.6794	NO
predicted_adaptive_train_win_count	73.1605	0.0781	1.6794	NO
predicted_adaptive120	73.1277	0.0802	1.6794	NO
predicted_arima120	73.1277	0.0802	1.6794	NO
predicted_knn120	73.1348	0.0798	1.6794	NO
predicted_multi_linearregression1_120	73.1558	0.0784	1.6794	NO
predicted_multi_linearregression2_120	73.2178	0.0743	1.6794	NO
predicted_multi_linearregression120	73.1718	0.0774	1.6794	NO

Wind Speed Day 1 (23rd February 2018)

Algorithm	Mean	Tobserved	Ttable	Significance
Actual	14.2994	---	----	-----
predicted_adaptive_train_win_avg	13.8047	0.1688	1.6794	NO
predicted_adaptive_train_win_count	13.8045	0.1689	1.6794	NO
predicted_adaptive120	13.7894	0.1741	1.6794	NO
predicted_arima120	13.7894	0.1741	1.6794	NO
predicted_knn120	13.6845	0.2107	1.6794	NO
predicted_multi_linearregression1_120	13.8047	0.1688	1.6794	NO
predicted_multi_linearregression2_120	13.8046	0.1688	1.6794	NO
predicted_multi_linearregression120	13.8047	0.1688	1.6794	NO

Wind Direction 1 (23rd February 2018)

Algorithm	Mean	Tobserved	Ttable	Significance
Actual	148.4042	---	----	-----
predicted_adaptive_train_win_avg	165.6956	-0.5272	1.6794	NO
predicted_adaptive_train_win_count	165.7021	-0.5274	1.6794	NO
predicted_adaptive120	165.6383	-0.5256	1.6794	NO
predicted_arima120	165.6383	-0.5256	1.6794	NO
predicted_knn120	164.6809	-0.4980	1.6794	NO
predicted_multi_linearregression1_120	165.7187	-0.5279	1.6794	NO
predicted_multi_linearregression2_120	165.7000	-0.5274	1.6794	NO
predicted_multi_linearregression120	165.6956	-0.5272	1.6794	NO

Rainfall Day 1 (23rd February 2018)

Algorithm	Mean	Tobserved	Ttable	Significance
Actual	0.5755	---	----	-----
predicted_adaptive_train_win_avg	0.5484	0.2312	1.6794	NO
predicted_adaptive_train_win_count	0.5484	0.2312	1.6794	NO
predicted_adaptive120	0.5423	0.2848	1.6794	NO
predicted_arima120	0.5423	0.2848	1.6794	NO
predicted_knn120	0.5389	0.3148	1.6794	NO
predicted_multi_linearregression1_120	0.5484	0.2312	1.6794	NO
predicted_multi_linearregression2_120	0.5485	0.2303	1.6794	NO
predicted_multi_linearregression120	0.5484	0.2312	1.6794	NO

Atmospheric Pressure Day 1 (23rd February 2018)

Algorithm	Mean	Tobserved	Ttable	Significance
Actual	97266.6234	---	----	-----
predicted_adaptive_train_win_avg	97276.4922	-4.8657e-04	1.6794	NO
predicted_adaptive_train_win_count	97276.7065	-4.9713e-04	1.6794	NO
predicted_adaptive120	97274.2912	-3.7805e-04	1.6794	NO
predicted_arima120	97274.2912	-3.7805e-04	1.6794	NO
predicted_knn120	97283.4511	-8.2964e-04	1.6794	NO
predicted_multi_linearregression1_120	97278.0472	-5.6323e-04	1.6794	NO
predicted_multi_linearregression2_120	97278.3913	-5.8019e-04	1.6794	NO
predicted_multi_linearregression120	97276.4922	-4.8657e-04	1.6794	NO

Luminosity Day 1 (23rd February 2018)

Algorithm	Mean	Tobserved	Ttable	Significance
Actual	3.1621	---	----	-----
predicted_adaptive_train_win_avg	3.1628	-0.0011	1.6794	NO
predicted_adaptive_train_win_count	3.1628	-0.0011	1.6794	NO
predicted_adaptive120	3.1591	0.0046	1.6794	NO
predicted_arima120	3.1591	0.0046	1.6794	NO
predicted_knn120	3.1641	-0.0030	1.6794	NO
predicted_multi_linearregression1_120	3.1628	-0.0011	1.6794	NO
predicted_multi_linearregression2_120	3.1629	-0.0012	1.6794	NO
predicted_multi_linearregression120	3.1628	-0.0011	1.6794	NO

Appendix A2: Tests conducted with schemes not selected for the adaptive algorithms.

In addition to the schemes that were used to formulate the adaptive algorithms, tests were also conducted on the following schemes that were not included due to their inferior performance:

1. Linear Regression
2. Moving Average
3. Neural Network with Windows 15 and 30
4. Non-Linear Regression
5. Polynomial Regression with Degree 2
6. Polynomial Regression with Degree 3

The performance of these schemes for the 7 measured parameters are shown in Figures A2.1 to A2.11.

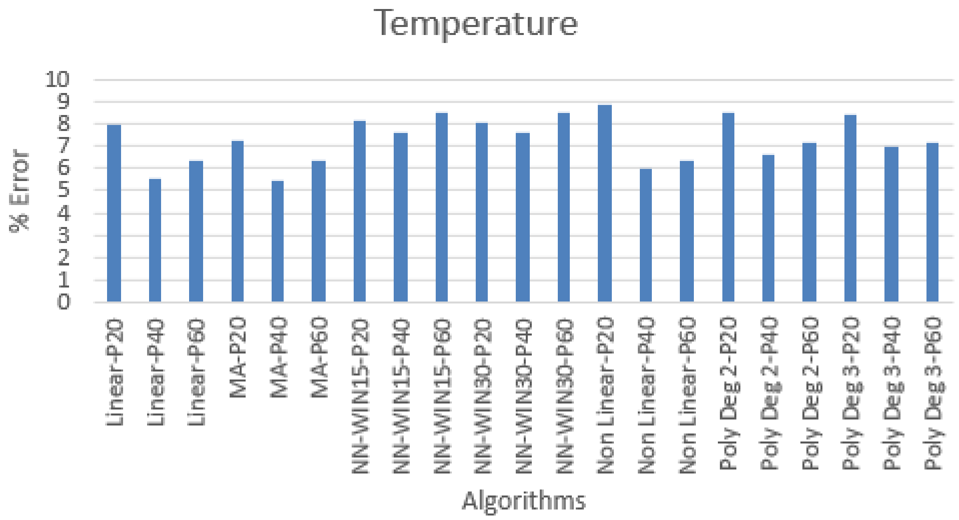


Figure A2.1: Percentage error for Temperature

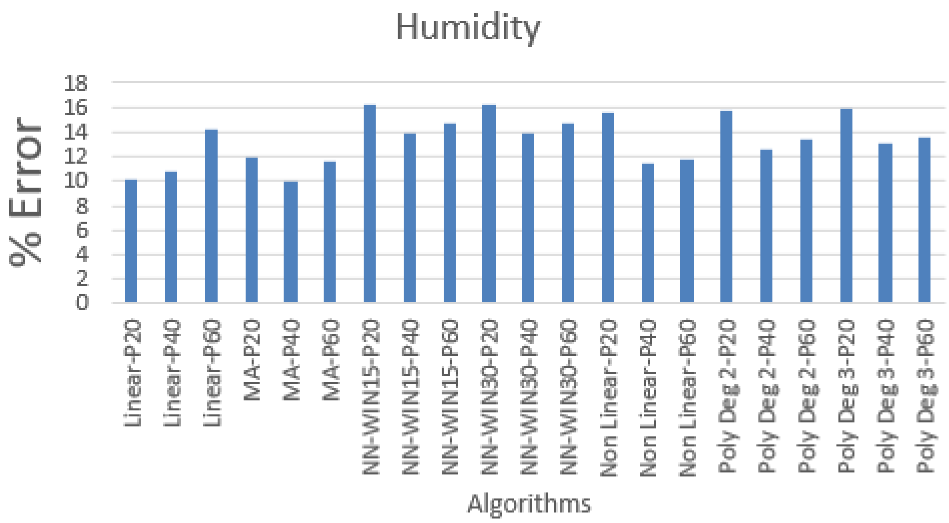


Figure A2.2: Percentage error for Humidity

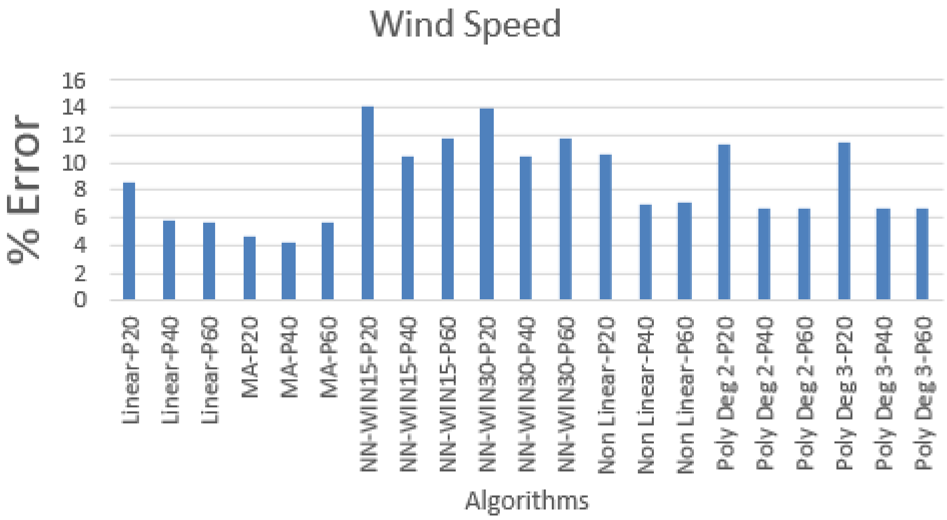


Figure A2.3: Percentage error for Wind Speed

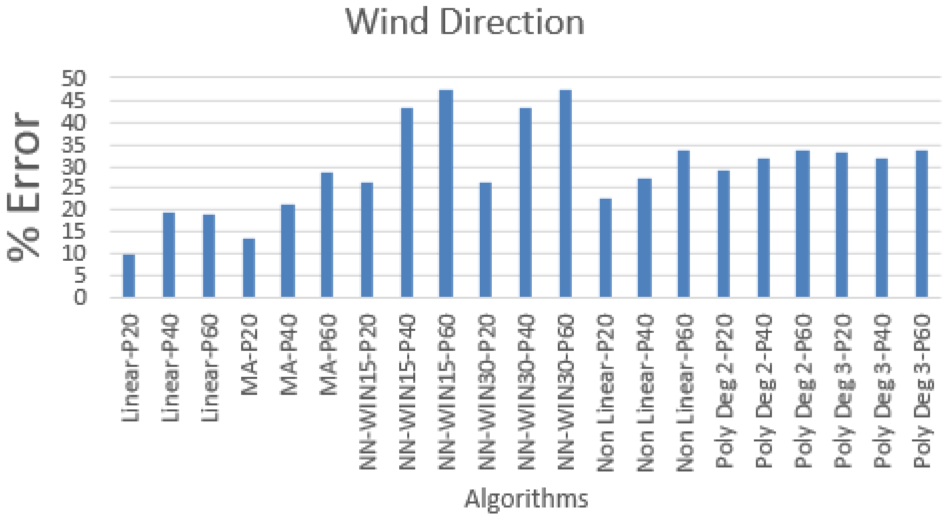


Figure A2.4: Percentage error for Wind Direction

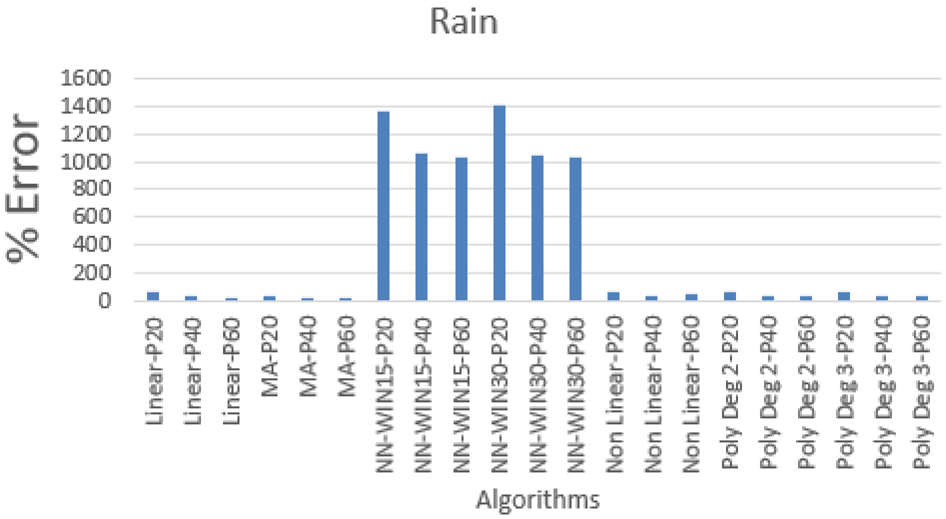


Figure A2.5: Percentage error for Rain

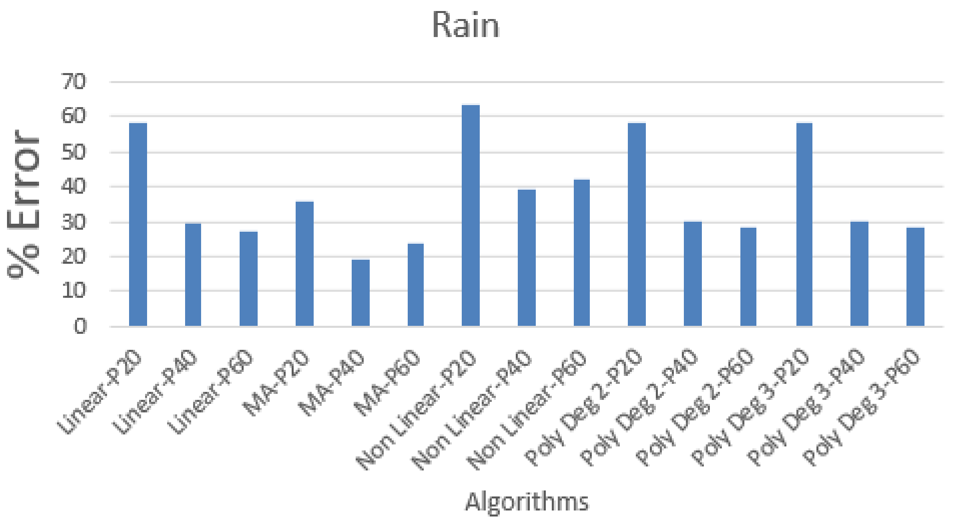


Figure A2.6: Percentage error for Rain

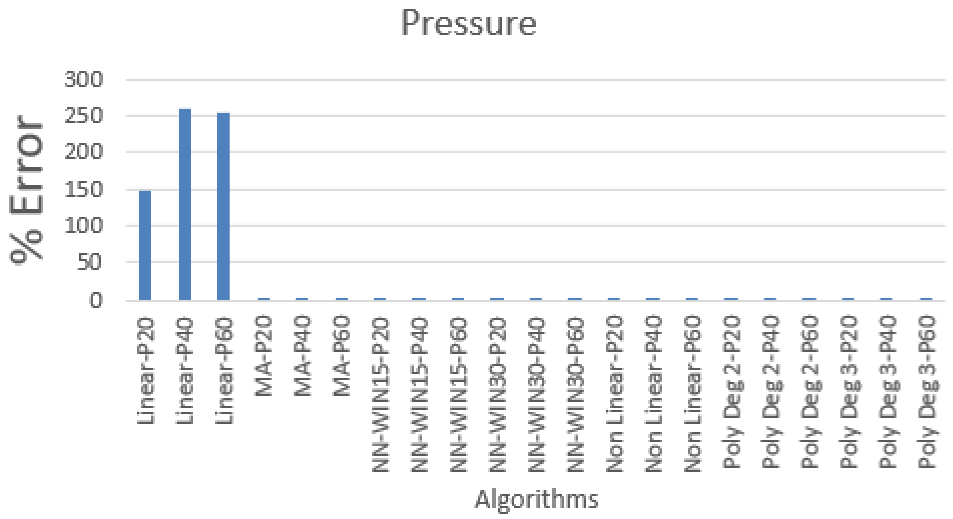


Figure A2.7: Percentage error for Pressure

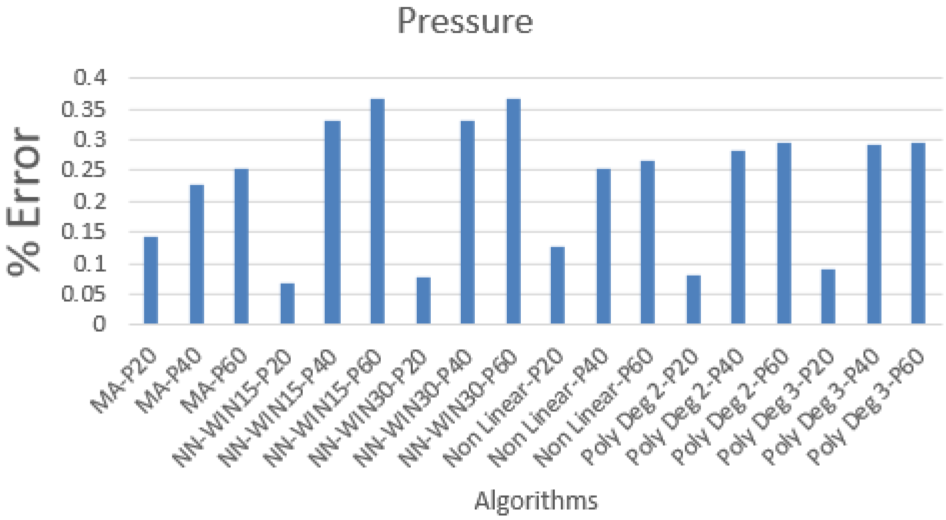


Figure A2.8: Percentage error for Pressure

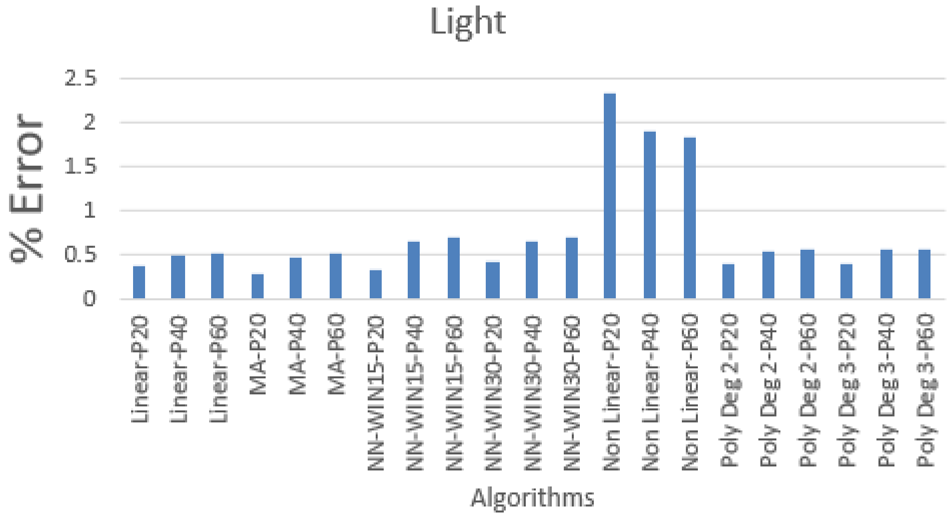


Figure A2.9: Percentage error for Light

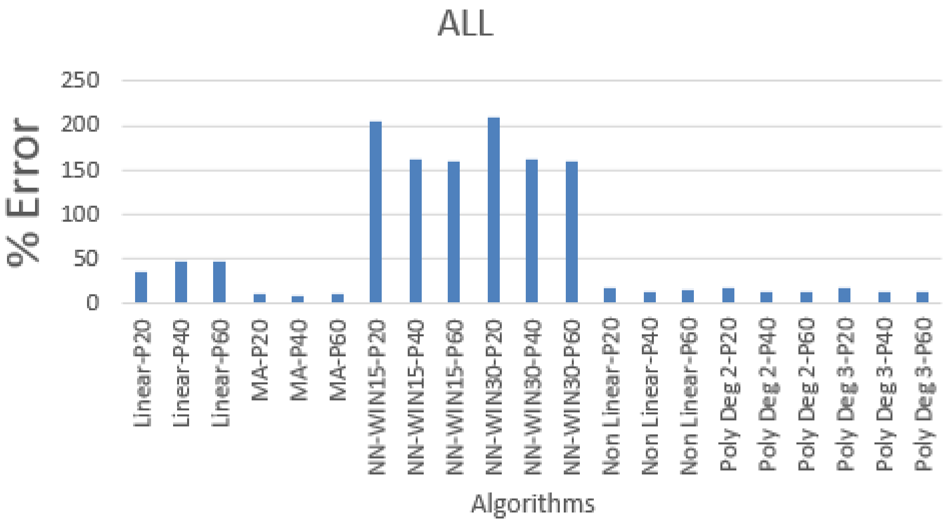


Figure A2.10: Overall percentage error

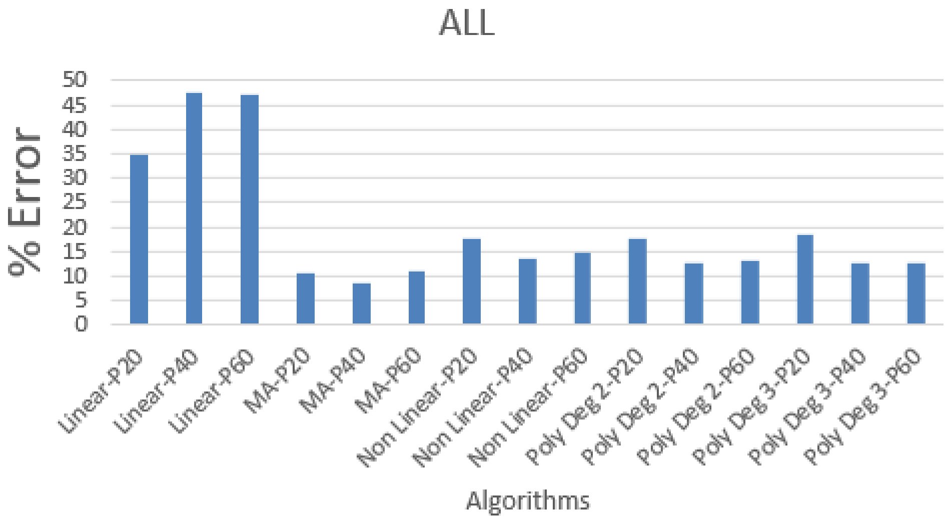


Figure A2.11: Overall percentage error