DETECTING ANOMALOUS SENSOR PLACEMENT THROUGH TEMPORAL ANALYSIS OF TEMPERATURE DATA

by

SUJEET VYANKATESH KULKARNI

(Under the Direction of Lakshmish Ramaswamy)

ABSTRACT

Crowdsensing temperature data has enabled a paradigm shift in the ways we collect data and analyze the heat exposure effects on individuals and small communities. Use of low-cost sensors has helped in gathering granular spatial-temporal temperature data and capturing ever-changing ambient environmental conditions. However, the practice poses challenges such as data integrity, and sensor failures. One of the main concerns is placement of temperature sensors such that they are shielded from the natural environment (for example, in air-conditioned vehicle, inside a bag) during data collection. This will lead to anomalous data collection. We propose a novel approach to detect anomalous sensor placement based on empirical observations, temperature readings of a sensor exposed to the natural environment show more frequent fluctuations than temperature readings of a sensor shielded from it. We use sliding window technique and supervised learning classifier to detect anomalous temporal temperature subsequences effectively. We also do comparative performance analysis of SVM, Logistic Regression and Random Forest classifiers.

INDEX WORDS: Anomaly Detection, Sliding Window, Zero-Crossing, Binary Classification, Crowdsensing, Time Series Data

DETECTING ANOMALOUS SENSOR PLACEMENT THROUGH TEMPORAL ANALYSIS OF TEMPERATURE DATA

by

SUJEET VYANKATESH KULKARNI

B.E., SHIVAJI UNIVERSITY, INDIA, 2005

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2018

DETECTING ANOMALOUS SENSOR PLACEMENT THROUGH TEMPORAL ANALYSIS OF TEMPERATURE DATA

by

SUJEET VYANKATESH KULKARNI

Major Professor:	Lakshmish Ramaswamy
Committee:	Thiab R. Taha
	John A. Miller

Electronic Version Approved:

Suzanne Barbour
Dean of the Graduate School
The University of Georgia
August 2018

DEDICATION

Thanks to my family members for their constant support and encouragement.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude for the continuous support and encouragement provided by Dr. Lakshmish Ramaswamy during my graduate academic career. I would also like to thank my committee members Dr. Thiab Taha and Dr. John Miller for their time and guidance.

I would like to thank Navid Hashemi, a PhD student at the University of Georgia for all his valuable inputs and support. Lastly, I would like to thank all my professors, friends, Department of Computer Science and University of Georgia for making my stay pleasant and memorable.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

INTRODUCTION

This chapter will discuss about the anomaly detection problem related to crowdsensed temporal temperature data. We have also discussed general opportunities and challenges related to crowdsensed data using low-cost sensors. In the last section we will succinctly discuss about our contribution and approach we are proposing to solve the problem.

1.1 Introduction

Applications such as monitoring weather changes, guiding agriculture activities, analyzing heat maps primarily rely on weather station and satellite generated data [3, 4]. Availability of low-cost small sensors connected to the internet using mobile devices are generating vast amount of new data. This provides an opportunity to develop new Internet of Things applications [1]. The data generated by sensors is granular in spatial and temporal aspects and provides opportunities for more dynamic and precise analysis. This has led to increased research activity on crowdsensed data.

Extreme individual heat exposure causes health issues and may lead to heat related illness like heat stroke, heat exhaustion [2]. Identification of such areas and guiding individuals appropriately can prevent such health hazards. Urban Heat Island (UHI) phenomenon observed in city areas is the result of heat generation because of urban infrastructure [4]. Heat maps generated based on weather station and satellite data are primarily used to analyze UHIs. This data fails to capture the dynamic nature [3] of environmental changes, and risk of individuals or small community's exposure to the heat. Crowdsensed temperature data provides an opportunity

to develop a system to analyze and guide individuals for heat exposure. However, crowdsensing temperature data using low-cost small sensors causes concerns about data integrity. One of the main concerns in crowdsensing temperature data is placement of sensor by the user [8]. Anomaly detection system needs to be designed to identify and filter the subsequences of temporal temperature data when temperature sensor was not exposed to the environment and was placed in a controlled environment to improve data integrity.

1.2 Crowdsensing

Increased availability of sensory devices like heart monitors, fitness trackers, air quality sensors, temperature, and humidity sensors are driving new era of Internet of Things applications [1]. These applications can be broadly classified into personal and community centric applications [1]. Applications collecting and analyzing sensory data related to tracking of individual physical movements, measuring health related parameters, can be classified as personal sensing applications [1]. Applications which deal with congregating spatial-temporal data from multiple individuals and analyze it to observe a wider community centric phenomenon are community centric applications [1]. Monitoring air quality of area, monitoring traffic are examples of community centric applications and they require either passive or active support of the multiple users [1].

1.3 Challenges in Crowdsensing

Crowdsensing though provides wide range of opportunities has its own challenges. Availability of limited energy resources, low computing powers, network bandwidth are specific areas of concern which need to be considered while designing a crowdsensing system [1].

Sensor failures, transmission errors, inconsistent system operation can affect data integrity. Crowdsensing temperature data helps in analyzing heat exposure of individuals and small

communities. Users may inadvertently keep the sensor in controlled environments like in the bag, in the pocket or in air-conditioned vehicles, and conduct the temperature recording experiment in the given area for heat exposure. The experiment, instead of recording the air temperature of the surrounding area will record temperature of the controlled environment and will affect the data integrity.

1.4 Thesis Contribution

Heat map generation and analysis using crowdsensed temperature data provides more granular, dynamic understanding of Urban Heat Island phenomenon. Our aim is to propose an anomalous sensor placement detection system which filters subsequences in temperature time series data when the sensor is placed in the controlled environment to improve data integrity.

Based on empirical observations, when a temperature sensor is exposed to ambient atmosphere shows more frequent fluctuations in the temperature readings compared to the readings of a sensor placed in controlled environments like pant pocket, air conditioning unit of vehicle, and inside the bag. To confirm the assessment, we conducted crowdsensing experiments and analyzed effects of air temperature, solar radiation, longwave radiation, and wind speed on readings of temperature sensor exposed to ambient atmosphere.

This study proposes features like zero-crossing point weight, non-zero temperature difference rate and applies concepts like zero-crossing rate [12], standard deviation to extract statistical features from subsequence of univariate temperature time series. Extracted features are used to define and identify patterns in the exposed and the unexposed temperature sensor readings and classify the temporal temperature subsequences accordingly.

The study does a comparative performance analysis of proposed anomaly detection approach with temperature time series data having different time intervals and window sizes.

Anomaly detection approach in the study uses rolling sliding window [10] for feature extraction and does the binary classification of subsequences in the temperature time series. Thus, identifying subsequences which form collective anomalies due to erroneous sensor placements. Sliding window approach can be used to map a subsequence of sequential data to a single output [11]. In our approach, we generate supervised learning models based on historical data and classify the temperature time series subsequences. The study gives a comparative analysis of supervised learning models like SVM, Logistic Regression, and Random Forest and proposes suitable classifier based on the performance analysis.

CHAPTER 2

BACKGROUND

2.1 Urban Heat Island (UHI)

Urban Heat Island (UHI) is identified by warmer air in the urban area surrounded by cooler air in rural or suburban areas [5]. Urban Heat Island is the result of factors like air-conditioning units, use of vehicles, industries, building structures, lack of vegetation, and decreased moisture [5]. The observations about it were done long time ago, in 1833 Luke Howard did temperature analysis in and around London and observed city areas were warmer than the surrounding rural areas [6].

UHI study is primarily based on study and evaluations of the heat maps generated for large city areas. Weather station data and satellite data is primarily used to generate the heat map of a city area [4]. Urban Heat Island phenomenon is observed because of both natural and human causes [4]. Lack of vegetation and water bodies, increased pollution, increased road and building infrastructure introduces dynamicity to UHI effect and exposes individuals to heat effects. Extreme heat exposure causes heat related illness based on physical condition of individual [2]. Analyzing dynamic nature of UHI effect due to ever changing ambient environmental conditions and its impact on individual will help in preventing heat related health hazards. Overall aim of the project is to design a system to collect granular spatial-temporal temperature data and generate heat maps to study heat exposure effects on individuals and small communities.

2.2 Crowdsensed Temperature Data

Temperature data is primarily gathered using sources like weather stations and satellites. Due to

the high costs involved in the weather station installations there are limited number of fixed weather stations covering a geographical area [7], and satellite rotations are periodic in nature. This results in coarse temperature data in spatial and temporal aspects. Therefore, the study of Urban Heat Islands based on remote sensing data lacks the granularity, dynamicity needed to study the effect of vegetation, building structures, local heat generating activities like use of motor vehicles, use of air-conditioning units on heat generation and corresponding heat exposure effects on individuals, small communities.

To supplement the existing ways of temperature data collection [7], and to cover the gaps in spatial and temporal aspects of temperature data, crowdsensing can be used as an effective technique. Temperature sensors used in crowdsensing temperature data are small and do not employ radiation shields as used in weather stations. They are designed in a way to be carried easily by the user on the backpack or on belt-loop to collect the data effortlessly.

Crowdsensed temperature data, due to its very nature pose challenges about the accuracy [7] and methodology incorporated to gather the data [8]. The proposed anomaly detection approach in this thesis attempts to provide a systematic solution to find and eliminate these anomalies.

2.3 Outlier Detection

Outliers are basically patterns in the data which do not confirm to normal behavior [9]. Outlier detection is an important step to clean the input data. It helps improve the reliability of the data and improves effectiveness of the analysis. Outlier detection techniques are employed to analyze the data in different domains like healthcare, oil and gas, finance, and weather forecasting [9].

Below given are different categories in which we can broadly classify the outliers,

- Point Outliers: A single data point is considered as point outlier if it does not fit in the pattern with regard to rest of the dataset [9]. A high value credit card transaction for a

user will be treated as an outlier in case all other transactions are low value [9].

- Contextual Outliers: A single data instance is considered as outlier in a specific context but not otherwise [9].

  In this case, each data point is defined using two attributes,

  1. Contextual Attribute: Contextual attributes are used to define the context of a data point [9]. Longitude and latitude determines the context in spatial dataset, and in temporal dataset, time is the contextual attribute [9].
  2. Behavioral Attribute: Behavioral attributes define the non-contextual attributes of a data point [9]. In temporal temperature dataset, temperature value of a data point is a behavioral attribute.

- Collective Outliers: A collection of related data points is outlier with respect to the entire data set, then that is identified as a collective outlier [9]. In a human electrocardiogram (ECG), if a low value exists for an extended amount of time then that can be classified as collective outlier [9].

2.4 Sliding Window for Time Series Data

Domain plays an important role in defining outliers in time series data. In certain time series outlier detection problems, properties of subsection of time series data can be more important than properties of entire time series [10]. As per definition of the outlier, either an entire subsection of time series can be classified as outlier, or a subsection can be used as input to predict the next value in time series to identify outlier. Therefore, technically all subsequences in time series can be classified as outliers or need to be extracted for prediction purposes. The sliding window algorithm helps to extract all such subsequences in a given time series [10].

Suppose a temperature time series is defined as,

$D = \{T_1, T_2, T_3, T_4, \ldots, T_k, T_{k+1}, \ldots, T_n\}$

Then a subsequence S1 having elements k can be defined as,

$S1 = \{T_1, T_2, T_3, T_4, \ldots, T_k\}$

As the window representing subsequence moves by offset 1 in the time series D new subsequence,

$S2 = \{T_2, T_2, T_3, T_4, ..., T_{k+1}\}$

By moving with offset 1 all the subsequences in a time series D can be extracted.

In a supervised learning problem, a classifier is constructed to predict classes in the test data by training it in on the historical data [11]. In case of time series data, the data consists of sequences and is not drawn independently and identically from joint distribution [11]. The data sequences show correlations and nearby values are related [11]. Therefore, the sequential data does not fit supervised learning framework properly [11]. Order in the sequential data is important and contains correlations, it can be used to improve the classifier performance [11].

Sliding window can be used to convert a window of sequential data S to a single output d, and then all the subsequences in time series can be mapped to respective values and a supervised learning algorithm can be applied [11]. Thus, with sliding window method we can apply conventional machine learning methods to solve sequential machine learning problems [11].

[10] Yu et al. uses sliding window method and k-nearest neighbor window prediction model for predicting next data point in the time series and identifying outliers in hydrological time series data. The method uses Prediction Confidence Interval (PCI) as threshold to identify outliers from observed values. PCI is calculated dynamically based on predicted value and confidence coefficient, which increases time complexity. Proposed approach in the study does not require labeled data and claims to improve outlier detection performance in hydrological time series.

2.5 Zero-Crossing

Zero-crossing is defined as change of sign or crossing of zero in a sequential data [13].

Zero-crossing rate [12] is total number of zero-crossings in a sequential data divided by total number of instances in that sequence.

Consider a series with starting index as 1,

D = {1, 1, 0, -1, 0, 1, 1, 0}

Zero-crossing indexes of series D are Z = {3, 5}

Zero-crossing rate = (Total instances in series Z) / (Total instances in series D).

Zero-crossing rate = 0.25.

Zero-crossing rate is used for event detection, pattern matching and classification in sequential data. Gouyon et al. uses zero-crossing rate for classification of percussive sounds [12]. The proposed framework uses non-supervised classification technique, Agglomerative Clustering, and zero-crossing rate as feature to classify snare drum like and bass drum like sounds from audio signal. Agglomerative clustering groups the initially created individual sound clusters based on distance, eventually forming required clusters [12]. The study claims to efficiently classify the two classes from audio signal, but its application is limited in scope and they have not applied it to large database.

In CHAPTER 3 we will discuss application of zero-crossing rate for our proposed anomaly detection approach.

2.6 Supervised Binary Classification

In supervised binary classification, a model is created based on labeled training data and is applied to classify unseen data points into two classes. Model learns patterns during training phase, which is then used for classification. In this section we will primarily discuss Support

Vector Machine, Logistic Regression and Random Forest classifiers.

2.6.1 Support Vector Machine (SVM)

SVM needs labeled training dataset to train the classifier model, which is then used to classify the given test dataset. In a binary classification given dataset has two classes, positive class and negative class. SVM uses multiple hyperplanes to linearly separate the positive and negative class data points in the feature space [15]. The one hyperplane which is placed at maximum possible distance from nearest positive and nearest negative class data points, and which maximizes boundary distance between the positive and negative class data points is chosen [15]. The positive and negative data points which lie on the corresponding boundaries with maximum margin are called support vectors [15]. Support vectors define boundaries of the classes in feature space and contain all the necessary information required to do the classification [15]. Linear kernel is used for classification of linearly separable class boundaries. If class boundaries are not linearly separable then nonlinear kernel functions like Radial Basis Function (RBF), polynomial, sigmoid are used to classify the data points. The kernel functions are used to map nonlinear data to higher dimensional feature space where linear classification of the data is possible [15]. The choice of kernel based on the data separation in feature space plays an important role in efficiency of SVM classifier. SVM classifier has extensive use in industry wide applications like spam categorization [14], bankruptcy prediction [16].

Min et al. uses Principal Component Analysis (PCA) to reduce the dimensionality of financial data and use different SVM kernel functions to compare their performance [16]. Based on performance analysis they choose SVM with RBF kernel function for bankruptcy prediction problem. Further they statistically compared SVM performance with Back-Propagation Neural Network (BPN), Multiple Discriminant Analysis (MDA) and Logistic Regression (LR) for the

financial data and found SVM outperformed the other models. The study primarily focuses on financial data and so performance evaluation is problem specific [16].

2.6.2 Logistic Regression (LR)

Logistic Regression is used to establish relationship between dichotomous dependent variable with continuous or categorical independent variables [17]. The core of Logistic Regression is a logit function. Dichotomous dependent variable has two outcomes, in binary classification it can represent a positive class by value 1 and a negative class by value 0. P is the probability of dependent variable having value 1 and (1 – P) represents probability of dependent variable having value 0. Y represents odds of dependent variable, it's the ratio of probability P of dependent variable to the (1 – P) [17, 18]. Independent input variables are represented by x_i.

The logit function [17, 18] can be given as,

$$\text{logit}(Y) = \text{In}(P/(1\text{-}P)) = b_0 + b_1 x_1 + \ldots + b_n x_n$$

$$\text{logit}(Y) = b_0 + \sum_{i=1}^{i=n} b x_i = b_0 + BX$$

In the above equation b_0 is an intercept and b_1 to b_n are the coefficients of corresponding independent variables from x_1 to x_n.

Therefore, the probability P of dependent variable can be given by below equation [17, 18],

$$P = 1 / (1 + e^{-(b0 + BX)})$$

Logistic Regression is a supervised learning classifier, labeled training data is needed to train the classifier. It is used in outlier detection, classification tasks. [18] Subasi et al. uses Multilayer Perceptron Neural Network (MLPNN) and Logistic Regression (LR) for classification of electroencephalograph (EEG) signals. They use Lifting-Based Discrete Wavelet Transform (LBDWT) technique to divide four-channel EEG signal into sub-bands and give it as input to LR

and MLPNN to classify the EEG data into normal and epileptic class. For evaluating the binary classification problem of EEG data, they use accuracy, sensitivity and specificity performance measures and noticed MLPNN performed better than LR using statistical analysis. Subasi et al. claims the proposed method improves computation efficiency and gives better classification performance. For evaluation purpose, study relies on sample EEG data validated by human experts and have not applied the technique to unseen EEG signals [18].

2.6.3 Random Forest (RF)

Random Forest builds multiple classification trees and uses majority vote to determine class of a given instance [19]. As mentioned in the paper the algorithm in the training phase selects many bootstrap samples such that in each bootstrap sample, 63% of original samples occur at least once. The training data samples which are not there in bootstrap samples are called out of bags samples. As discussed in the study a classification tree is then fit to a bootstrap sample and fully grown. Then multiple such classification trees are used to predict out of bags samples.

Random Forest is used for classification, regression and outlier detection tasks. [19] Cutler et al. uses it for the classification of ecological data. The study provides comprehensive analysis of RF and its applications. As discussed in the study ecological data is high dimensional, with nonlinear complex relations between variables and contains missing values. Therefore, use of linear statistical methods is limited and Random Forest (RF) is more suitable for the purpose. To cover wider range of ecological data and their relations, they used three different ecological examples and applied them as input to compare RF performance with Logistic Regression, Linear Discriminant Analysis (LDA), Additive Logistic Regression and Classification trees. The study used statistical measures like sensitivity, specificity, kappa and Percentage Correctly Classified (PCC) for performance comparison. They found higher RF performance in the category of

examples where complex nonlinear relationship between variable was involved and moderate superiority of RF was observed otherwise. The study primarily focuses on ecological data to establish RF performance superiority in classification of dataset, with non-linearly related variables, and needs to be extended to datasets from different domains.

CHAPTER 3

OVERVIEW

3.1 System Architecture

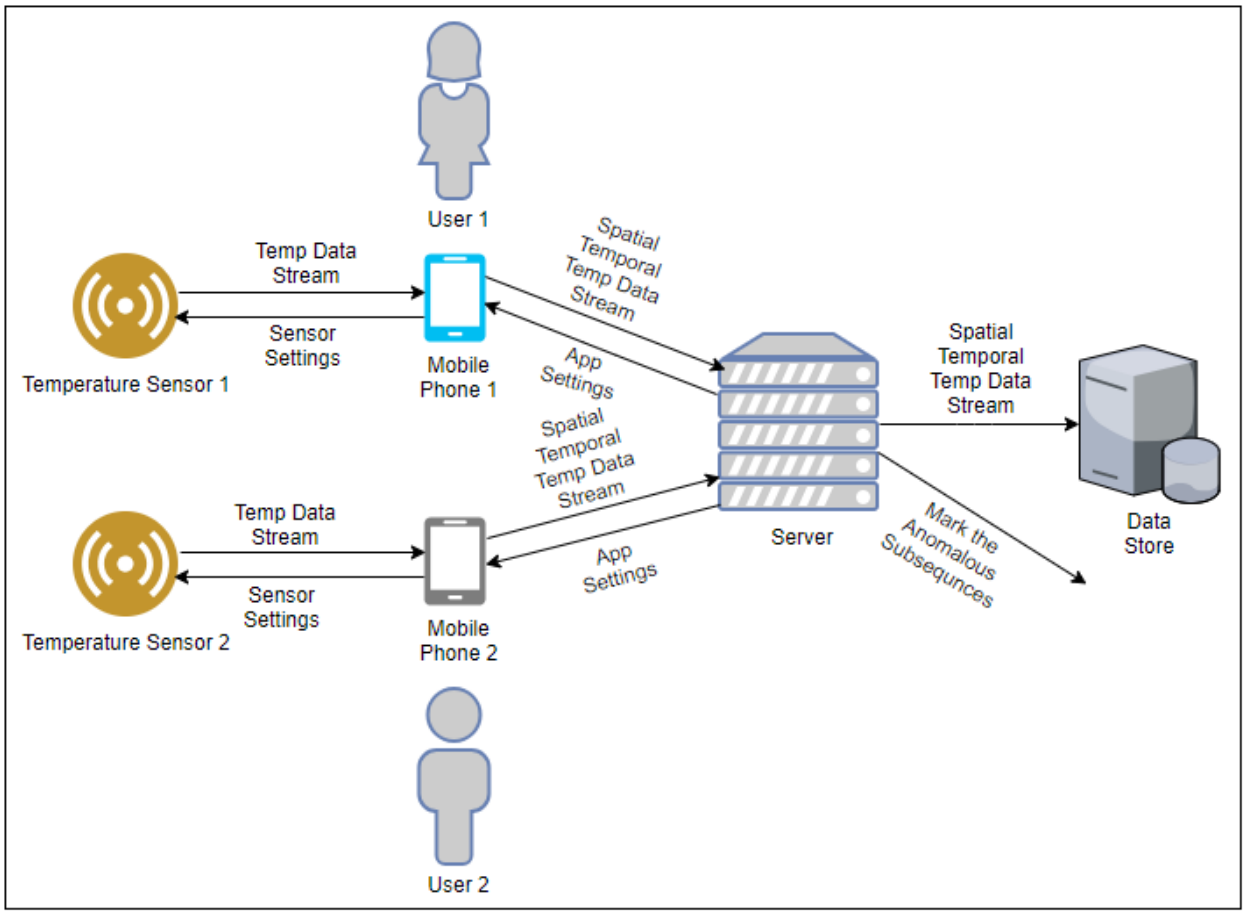

Figure 1: High level system architecture

Above diagram shows high level architecture of crowdsensed data collection using temperature sensors. Our system utilizes temperature sensors available in the market to collect the granular temperature data in a given area. For our primary research we have used Kestrel DROP Sensors. The system architecture is distributed in nature and can be extended to support different

temperature sensors via Bluetooth protocol. Sensors can be carried by user while walking or mounted on the vehicle for data collection. Mobile application acts as an intermediary and communicates with both sensor and server. Bluetooth protocol is used for sensor-mobile communication and mobile-server communication is done using web service over the internet. Temperature readings are synchronized with GPS data on the mobile based on timestamp, and then the spatial-temporal temperature readings are uploaded to the server. As can be seen in the diagram multiple users can simultaneously collect temperature data and upload it to the server. Server sends mobile application settings like data logging rate, which, in turn are communicated to sensor using Bluetooth protocol. Anomaly detection system is used to mark anomalous subsequences and filtered spatial-temporal temperature data is stored for further analysis.

3.2 Kestrel DROP Sensor

We primarily used Kestrel DROP sensors for temperature data collection. Kestrel DROP 2 has multiple inbuilt sensors to record meteorological measurements like Temperature, Relative Humidity, Heat Stress Index and Dew Point. Kestrel DROP 3 has additional sensor to record Station Pressure and Density Altitude. Figure 2 shows Kestrel DROP 2 sample readings and Figure 3 shows Kestrel DROP 3 sample readings.

Kestrel DROP sensors record temperature readings with (+/-) 0.5 ^{o}C accuracy, 0.1 ^{o}C resolution and specification range is -10 ^{o}C to 55 ^{o}C [20].

Mobile application communicates with Kestrel DROP sensors over Bluetooth protocol and user can download sensor data, monitor live readings, control the sensor settings like data logging rate using mobile application. The sensors are small and can be carried easily by the user for temporal temperature data collection.

FORMATTED DATE-TIME	Temperature	Relative Humidity	Heat Stress Index	Dew Point
YYYY-MM-DD HH:MM:SS	Â°C	%	Â°C	Â°C
9/10/2017 12:23	21.8	44.1	21	9
9/10/2017 12:23	21.1	42.3	20.2	7.8
9/10/2017 12:23	20.8	41.8	19.6	7.3
9/10/2017 12:23	20.5	41.1	19.1	6.8
9/10/2017 12:23	20.2	39.9	18.7	6.2
9/10/2017 12:23	20.3	39.7	18.7	6.2
9/10/2017 12:23	20.2	39.5	18.6	5.9
9/10/2017 12:23	20	38.9	18.3	5.6

Figure 2: Kestrel DROP 2 sample readings

FORMATTED DATE_TIME	Temperature	Relative Humidity	Heat Stress Index	Dew Point	Station Pressure	Density Altitude
YYYY-MM-DD HH:MM:SS	Â°C	%	Â°C	Â°C	mb	m
8/4/2017 19:13	28.6	70	32.5	22.6	992.7	799
8/4/2017 19:13	28.7	69.7	32.5	22.6	992.8	798
8/4/2017 19:13	28.7	69.4	32.7	22.6	992.7	799
8/4/2017 19:13	28.7	69.3	32.7	22.5	992.7	799
8/4/2017 19:13	28.7	69.3	32.7	22.5	992.7	799
8/4/2017 19:13	28.8	69.3	32.7	22.6	992.7	801
8/4/2017 19:13	28.8	69.4	32.7	22.6	992.7	801
8/4/2017 19:13	28.8	69.5	32.9	22.7	992.7	803

Figure 3: Kestrel DROP 3 sample readings

3.3 Empirical Observations

We collected temperature data to study the impact of temperature sensor placements on the temperature sensor readings and to confirm our observations, temperature readings of the sensors exposed to the ambient atmosphere show more frequent fluctuations compared to the temperature readings of the unexposed sensors.

The data was collected from August 2017 to October 2017 by 4 volunteers using 12 Kestrel DROP sensors. To avoid any bias, data samples were collected from morning to the evening on different dates over a span of 3 months. Total we have collected 131 data samples, totaling 77 hours of temperature data and have used 5 seconds data logging rate during collection. The data

collection involved temporal temperature reading samples from vehicle mounted, user borne and stationary sensors. To understand the effects of sensor placements on the readings during data collection, temperature sensors were exposed to the environment and were also kept in closed or controlled environment, unexposed to the ambient atmosphere.

The data collection was done following standard guidelines related to sensor placement and data logging rate; and time stamps were noted in order to label the collected data. The data gathered in controlled environment was labeled as Class '1' and the data gathered by keeping the sensor exposed to the environment was labeled as Class '2'.

Lable/Class ↓	User/Sensor Movements → (Travelling Mode) / Sensor Position ↓	Stationary	Walking	Bus	Car	Total	
1 (Closed space/Controlled Environment)	stationary_in_ac_home	6				6	91
	walking_in_bag		14			14	
	walking_in_pocket		16			16	
	in_ac_car				13	13	
	in_ac_car_in_pocket				7	7	
	in_ac_car_in_bag				7	7	
	in_ac_bus_on_bag			5		5	
	in_ac_bus_on_belt_loop			5		5	
	in_ac_bus_in_pocket			9		9	
	in_ac_bus_in_bag			9		9	
2 (Exposed to Environment)	stationary_outside_in_sun	8				8	40
	stationary_outside_in_shadow	6				6	
	walking_on_belt_loop		8			8	
	walking_outside_bag		9			9	
	outside_car				9	9	
	Total	20	47	28	36	131	

Figure 4: Data collection for multiple subcategories

We have further subcategorized the data collection based on travelling mode and sensor placement to cover the different possible erroneous situations in which temperature data could be collected during crowdsensing and uploaded to the server. Figure 4 shows data collection for all 131 samples. The subcategorization helped us to understand possible common patterns and

variations in temporal temperature readings, which can be used for anomaly detection.

3.4 Empirical Analysis of Crowdsensed Temperature Readings

We performed empirical analysis on the collected temperature data based on subcategories discussed above to identify, define patterns for anomaly detection. It involved visualization and comparative study of temporal temperature sequences collected by different users under same or different categories. In this section, we discuss experiments conducted by different users on different dates and have analyzed exposed and unexposed sensor readings. We have also discussed experiments showing deviation from normal observation.

3.4.1 Car and User Walking Experiment

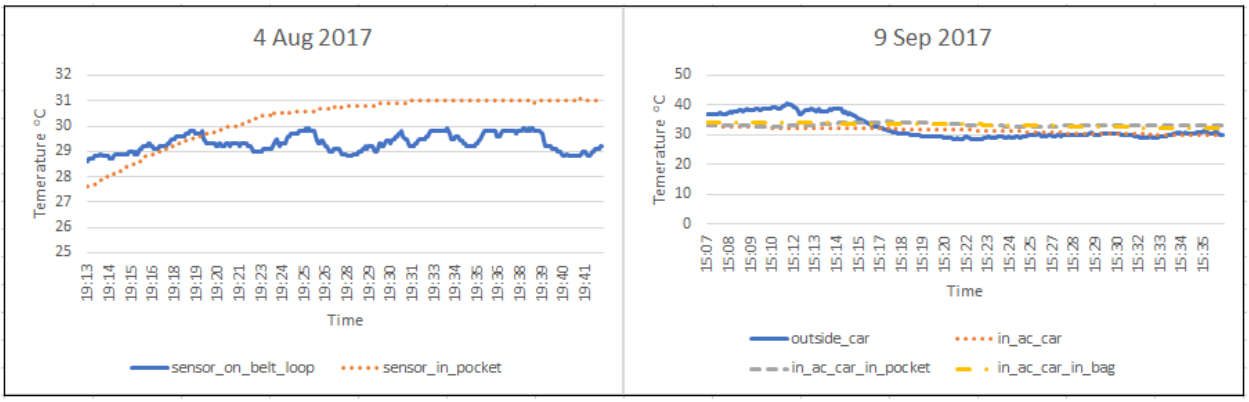

Figure 5: Temperature readings on 4 Aug 2017 and 9 Sep 2017

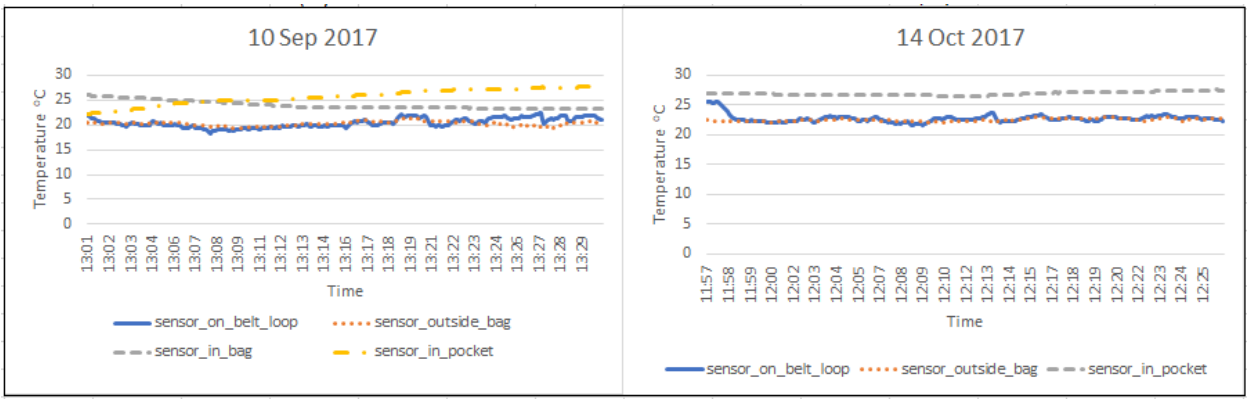

Figure 6: Temperature readings on 10 Sep 2017 and 14 Oct 2017

In the experiment conducted on 10 Sep 2017 shown in Figure 6, user carried 4 Kestrel DROP sensors. Two temperature sensors are exposed to the outside environment, one is placed on the user's belt-loop and other one is placed on user's bag. Other two temperature sensors are unexposed to the ambient atmosphere, one is placed in user's pant pocket and other one is placed in the user's bag. As can be seen in Figure 5 and in Figure 6, readings from the sensors which are not exposed to the outside environment are different from the sensors exposed to outside environment. Temperature readings of the unexposed sensor are controlled by the environment they are placed in and are not affected by changing surrounding environmental conditions, therefore show less fluctuations.

To confirm our observation, we have collected 131 data samples and observed similar patterns. Above figures show, results from the other such experiments conducted on different dates. Body heat, material of bag and inside temperature of the bag are influencing the temperature readings of unexposed sensors, and their readings to a certain extent are independent of outside air temperature. In the crowdsensing experiment to collect the temperature data user, vehicle moves geographically with the sensor and sensor readings change based on changed surroundings. Also, readings of the temperature sensors exposed to the environment are affected by surrounding air temperature, longwave radiation, solar radiation [21] and thus show fluctuations. Weather stations deploy temperature sensors at fixed locations at certain height from the ground and use radiation shields [22, 23] to reduce the effects of solar radiation, longwave radiation on temperature sensors. Therefore, the fluctuations though observed are less frequent in the weather station temperature data. Structural patterns observed in the crowdsensing experiments between exposed and unexposed temperature sensor readings can be used to identify temperature sensor placements and effectively classify corresponding subsequences.

3.4.2 Bus and Stationary Sensor Experiment

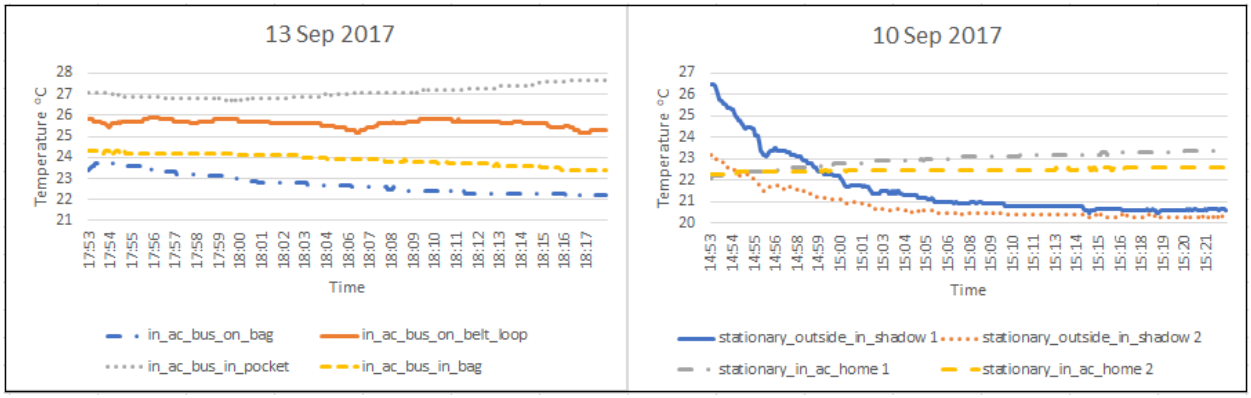

Figure 7: Temperature readings on 13 Sep 2017 and 10 Sep 2017

In the temperature data collection experiment shown in Figure 7, we can see that on 13 Sep 2017 multiple sensors are carried by the user travelling in an air-conditioned bus and temporal temperature readings of sensors belong to the controlled or unexposed category and are labeled as Class '1'. However as can be seen in the Figure 7, certain subsequences of temperature sensor readings placed on belt-loop shows fluctuations and may get classified as exposed. In this case a sitting user travelling in an air-conditioned bus is carrying the sensor on belt loop, therefore the temperature sensor is getting regulated by heat sources like body heat and air-conditioned unit, causing fluctuations. In the second experiment shown in Figure 7 conducted on 10 Sep 2017, sensors are placed stationary in shadow of a porch. Sensors are exposed to outside environment and sensor readings are labeled as Class '2'. However, as sensors are stationary and not exposed to solar radiation certain subsequences in the temporal temperature readings do not show pattern of fluctuation and may get classified as unexposed and marked as anomalous.

Data collection experiment discussed in Figure 7, helps us to understand that certain subcategories of temperature data collection may show patterns which are contrary to the normal observed behavior in exposed and unexposed temporal temperature sensor readings.

3.5 Generating Temporal Temperature Data with Different Time Intervals

Data logging rate of the temperature sensors is an important aspect in crowdsensed data. It affects the sensor battery life, and in case of a non-stationary sensor affects granularity of the temperature readings for a given geographical area. Temperature sensors allow users to set different data logging rates like 5, 10, 20, 30, 60 seconds and based on which temperature readings are recorded by the sensors. We have consistently used 5 seconds data logging rate during temperature data collection experiments. To study the impact of data logging rates on our anomaly detection approach, we have generated temperature time series with 10, 20, 30 and 60 seconds time intervals by selecting corresponding instantaneous values from the original temperature time series with 5 seconds interval.

3.6 Defining Window Size and Offset

Defining window size and offset is crucial in identifying anomalous subsequences in time series temperature data. Based on empirical analysis of the collected temperature data, we have used two, 5 minutes and 10 minutes, window sizes and rolling window offset of 1. Window size affects feature extraction and impacts the accuracy of the classifier. Defined window sizes contain sufficient temperature readings forming subsequences of appropriate size and are small enough to do almost real time anomaly detection. Rolling over window with the offset 1 was used to maximize the training and testing dataset samples. This helps in capturing all possible subsequences and their variations in the historical data, creating a well-trained classifier; and results in identification of all the anomalous subsequences in the streaming data.

3.7 Feature Selection

Temporal temperature data is a univariate time series and we extract statistical features of given subsequence to train the classifier. Feature extraction approach similar to training phase is used

to classify patterns from the temperature data stream. Consider a temporal temperature time series with starting index as 1,

D = {24, 25, 24, 24, 23, 25, 25, 26}

Then from temperature series D, series of consecutive temperature differences with starting index as 1 can be defined as below,

C = {1, -1, 0, -1, 2, 0, 1}

Subsections below use series D and C stated above to define individual features, and they also discuss importance of the specific feature in improving classification accuracy.

3.7.1 Standard Deviation of Consecutive Temperature Differences (StdDevTempDiff)

Standard deviation gives us an idea of how distributed observations are around the mean [24]. High value of standard deviation of consecutive temperature differences, series C, in given window gives us idea about high variations in the temperature readings; which is an indication of sensor being exposed to the outside environment and not placed in a controlled environment.

Standard Deviation is calculated as,

$$SD = \sqrt{\left(\sum_{i=1}^{N}(X_i - M)^2\right) / (N-1)}$$

Where X_i is i^{th} value, M is the mean, and N is total number instances in series C.

3.7.2 Zero-Crossing Rate (ZeroCrossRate)

Zero-crossing [13] points in the consecutive temperature difference series ‘C’ are points where consecutive temperature differences change sign. This indicates effect of changes in surrounding conditions like solar radiation, wind speed, air temperature and longwave radiation on the temperature sensor readings.

Zero-crossing rate [12] is the number of observed zero crossing points to the total number of

consecutive temperature differences in the series.

Zero-crossing rate for a given subsequence indicates changes in surrounding environment and high zero-crossing rate is an indication of higher fluctuations in the temperature time series. As discussed in empirical analysis above, when visualized temperature time series from the sensors exposed to outside environment show high fluctuations.

Zero-crossing indexes of series C are Z = {1, 4}

Zero-crossing rate will be total instances in series Z divided by total instances in series C.

Zero-crossing rate = 2/7 = 0.28

3.7.3 Standard Deviation of Zero-Crossing Point Weights (ZeroCrossWtStdDev)

In certain cases, sensor readings placed in controlled environment might show fluctuations for small amount of time but will remain unchanged otherwise. Air conditioning units cause air circulation and temperature sensor might get exposed to air-packets with different temperature, affecting temperature readings. We can see this behavior in the experiment conducted on 10 Sep 2017 in the Figure 7. Temperature readings indicated by "stationary_in_ac_home 2" show fluctuations around 15:15 hours in the graph. This results in high zero-crossing rate for the subsequence and such subsequences may get wrongly classified as exposed. In this case valleys and peaks of fluctuations are (+/-) 0.1 ^{o}C, resolution of Kestrel DROP sensors [20].

We define zero-crossing point weight as the difference between temperature reading at the zero-crossing point and the mean of the temperature readings in the given subsequence.

Temporal temperature subsequence is then reconstructed by replacing temperature readings at zero-crossing points with corresponding zero-crossing point weights and other temperature readings are replaced with zero. The score indicating zero-crossing point weights in the subsequence is then calculated by taking standard deviation of the reconstructed subsequence.

Based on series ‘C’, ‘D’ and ‘Z’ defined above, series containing zero-crossing point weights can be defined as below,

Mean of series D = 24.5

Zero-crossing point indexes of series D are T = {2, 5}

Series of zero-crossing point weights W = {0, 0.5, 0, 0, -1.5, 0, 0, 0}

The feature value is standards deviation of values in series W.

High value of standard deviation of zero-crossing point weights indicates the fluctuations in temperature time series show high peaks and low valleys. The feature helps in reducing total number of false negatives.

3.7.4 Non-Zero Temperature Difference Rate (NonZeroTempDiffRate)

The feature indicates total number of non-zero temperature differences in consecutive temperature difference series divided by total number instances in the series.

From series ‘C’ defined above total number of non-zero temperature differences = 5

Total instances in series ‘C’ = 7

Therefore, non-zero temperature difference rate = 5/7 = 0.71

In certain cases when the temperature is either increasing or decreasing in particular direction, exposed temperature sensor readings show less fluctuations. Therefore, the corresponding subsequence may get wrongly classified as unexposed. High value of non-zero temperature difference rate indicates frequent changes in temperature values. This feature helps in reducing total number of false positives.

3.7.5 Sample Feature Data

For training the classifiers, feature extraction module extracts the features using sliding window technique from crowdsensed temporal temperature data and converts the subsequences into data

points. Individual data points in sample data were labeled as unexposed (Class '1') or exposed (Class '2') as per the sensor placements during data collection experiment.

StdDevTempDiff	ZeroCrossRate	ZeroCrossWtStdDev	NonZeroTempDiffRate	Class
0.022678605	0.033898305	0.064320998	0.050847458	1
0.022678605	0.033898305	0.064022418	0.050847458	1
0.022678605	0.033898305	0.063723868	0.050847458	1
0.022678605	0.033898305	0.063425347	0.050847458	1
0.022678605	0.033898305	0.063126856	0.050847458	1
0	0	0	0	1
0	0	0	0	1
0	0	0	0	1
0.206753311	0.305084746	0.329360963	0.745762712	2
0.206753311	0.305084746	0.328253209	0.745762712	2
0.206753311	0.305084746	0.327148958	0.745762712	2
0.200423281	0.288135593	0.303407159	0.745762712	2
0.199106713	0.271186441	0.288506686	0.745762712	2
0.199502596	0.271186441	0.287482394	0.762711864	2
0.199502596	0.271186441	0.285953565	0.762711864	2
0.196074867	0.271186441	0.284433998	0.762711864	2
0.195642179	0.254237288	0.241468846	0.745762712	2
0.19424809	0.254237288	0.241468846	0.745762712	2
0.190357871	0.254237288	0.239986979	0.745762712	2

Figure 8: Sample feature data

3.8 Training Classifiers for Binary Classification

In order to perform the binary classification, different supervised learning models for a given window size and time series interval were developed based on historical data. To generate the training and testing data sets, window sizes of 5 minutes and 10 minutes were used for extracting features from temporal temperature sequences with 5, 10, 20, 30 and 60 seconds of time intervals. We have used Logistic Regression, Support Vector Machine (SVM) and Random Forest classifiers to do comparative analysis of anomaly detection approach's performance.

We have developed a feature extraction module using pandas package [25] for python. The

feature extraction module, based on the window size and offset, chooses a subsequence in the temperatures series and extracts statistical features to generate the data sets used for classification. For creating trained supervised models, classifiers in “scikit-learn” package [26] for python were used.

Linear kernel with standardized input data was used to train the SVM classifier. Based on analysis of underlying data distribution, to avoid overfitting and to reduce computational complexity, we used linear kernel to train SVM. Nonlinear kernel functions map data points to higher dimensional feature space to achieve linear separability [15], increasing computational complexity. Therefore, making them unsuitable to be used in lightweight anomaly detection system. For training Logistic Regression classifier, training data was standardized and l2 penalty parameter was used. In case of Random Forest classifier, 100 trees were used as estimators and split quality was measured based on “gini” criterion.

The details of performance comparison between SVM, Logistic Regression and Random Forest will be discussed CHAPTER 4.

3.9 Anomaly Detection Approach

The study primarily focuses on designing an anomaly detection system aimed to identify the anomalous subsequences in temperature time series $\{T_1, T_2, \ldots, Tn\}$. For this purpose, we used rolling sliding window approach to convert the time series anomaly detection problem into binary classification problem. The system uses supervised learning model to identify and classify anomalous subsequences. Our method consists of following steps:

1. Based on the data logging rate of temperature sensor and size of sliding window, extract the subsequence from the temperature data stream.
2. Features such as zero-crossing rate [12], standard deviation of consecutive temperature

differences, standard deviation of zero-crossing point weights and non-zero temperature difference rate are extracted.

3. Supervised learning model classifies the subsequences based on the extracted features.
   a. If the subsequence is classified as anomalous, then it is marked and filtered out.
   b. Otherwise, the data stream is stored in the data store for further analysis.
4. Based on the defined offset, in our case set to 1, window is moved forward. Then the next subsequence is extracted from the temperature data stream for classification.
5. The process of anomaly detection continues till the end of temperature data stream.

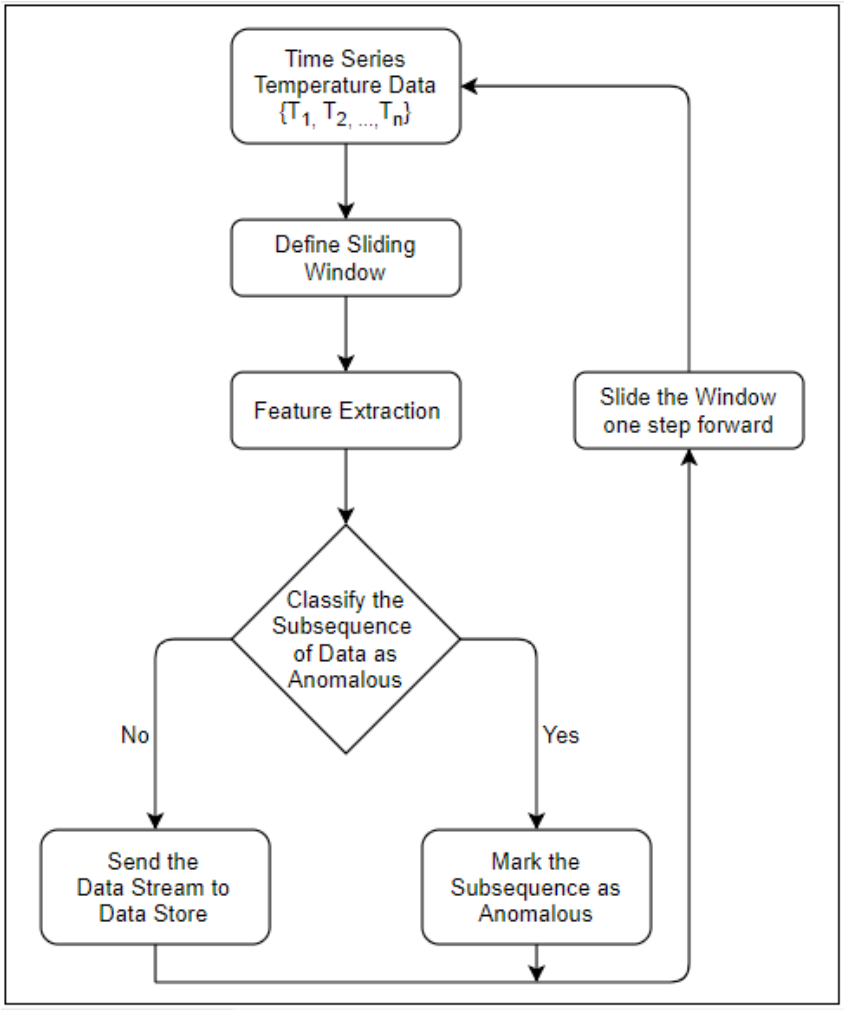

Figure 9: Anomaly Detection Technique based on Sliding Window

CHAPTER 4

EXPERIMENTS AND RESULTS

4.1 Data Distribution

Understanding of class distribution, feature wise class distribution and class boundaries is important to design a binary classification based anomaly detection system. The section uses graphs to provide an overview underlying distribution in the dataset. Temporal temperature datasets with 5 seconds interval, gathered in the data collection phase, were used for the analysis. The methodology and subcategories used to collect the data, and corresponding objectives and empirical observations are explained in CHAPTER 3.

Figure 10: Class distribution across dataset

Above figure shows class distribution in datasets, generated from 5 seconds interval temperature time series with 5 minutes and 10 minutes sliding window. Class '1' represents data points related to unexposed sensors and Class '2' represents data points related to sensors exposed to the ambient atmosphere. We have collected total 131 data samples with 5 seconds data logging rate. In temperature time series with 5 seconds interval, there are total 60 temperature readings in 5

minutes window and 120 temperature readings in 10 minutes window. We extract statistical features from the given window of time series and map them to a data point to generate the dataset. Total 48017 data points are present in 5 minutes window dataset and total 40157 data points are present in 10 minutes window dataset.

As seen in Figure 4 from CHAPTER 3, 10 subcategories were used for collecting unexposed class data samples and 5 subcategories were used for collecting exposed class data samples. Therefore, the unexposed class data samples are over represented than exposed class data samples in the dataset.

4.1.1 Feature wise Class Distributions

Our anomaly detection system extracts StdDevTempDiff, ZeroCrossRate, ZeroCrossWtStdDev and NonZeroTempDiffRate features from the sliding window to identify anomalous subsequences in the temperature series. Values of the extracted features depends on the size of sliding window and requires use of different supervised model for classification. This section helps us understand feature wise class distribution in the datasets, generated using 5 minutes and 10 minutes window on temperature time series with 5 seconds interval. We have used non-standardized data for generating graphs.

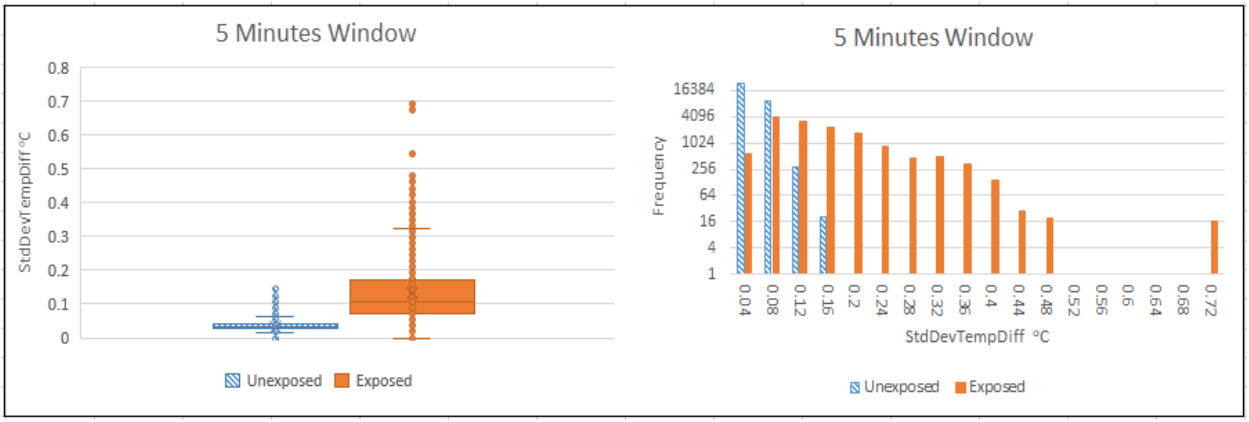

Figure 11: Class distribution for StdDevTempDiff feature in 5 Minutes Window dataset

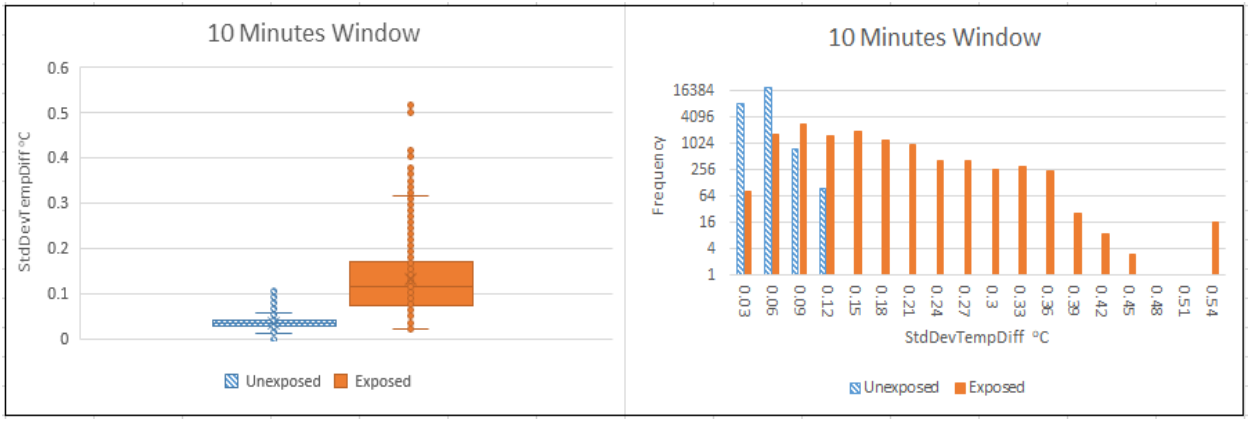

Figure 12: Class distribution for StdDevTempDiff feature in 10 Minutes Window dataset

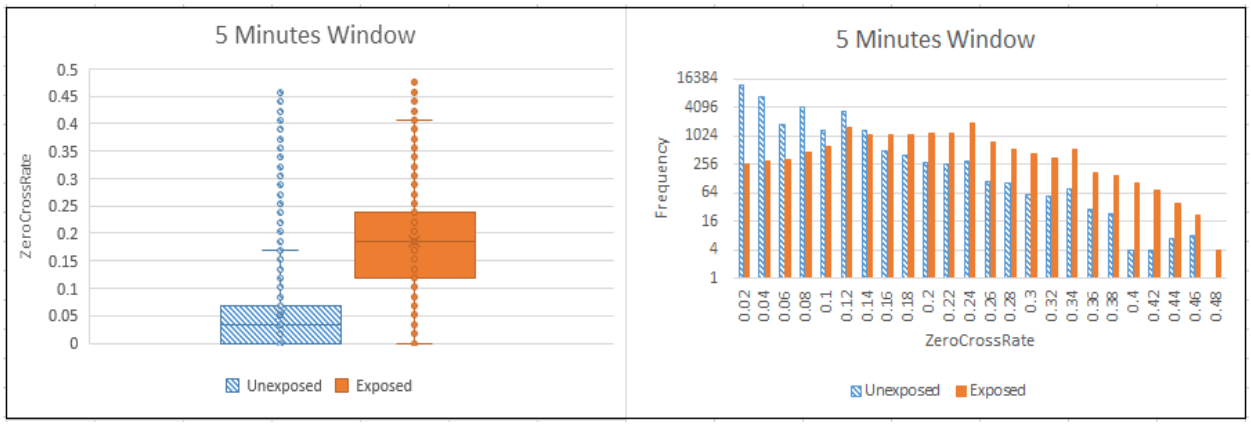

Figure 13: Class distribution for ZeroCrossRate feature in 5 Minutes Window dataset

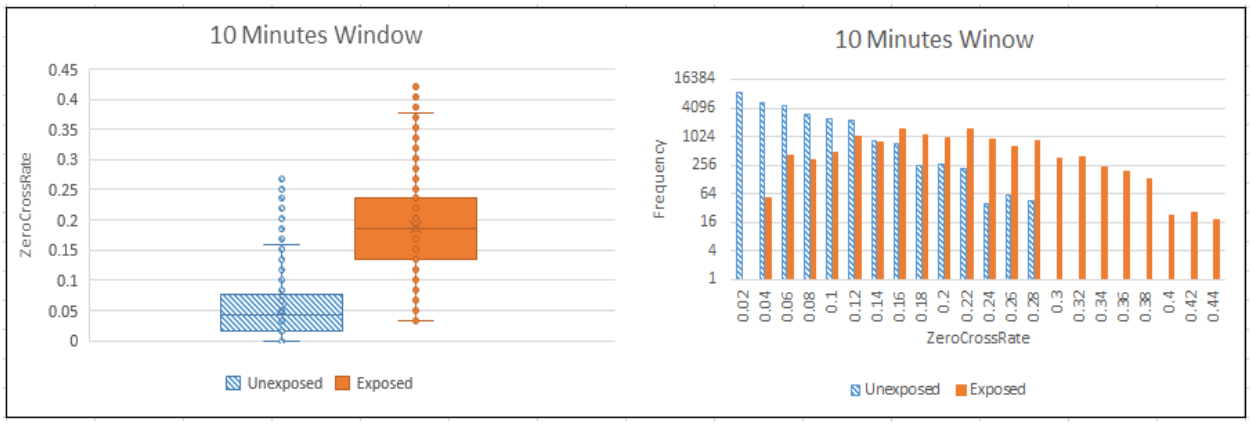

Figure 14: Class distribution for ZeroCrossRate feature in 10 Minutes Window dataset

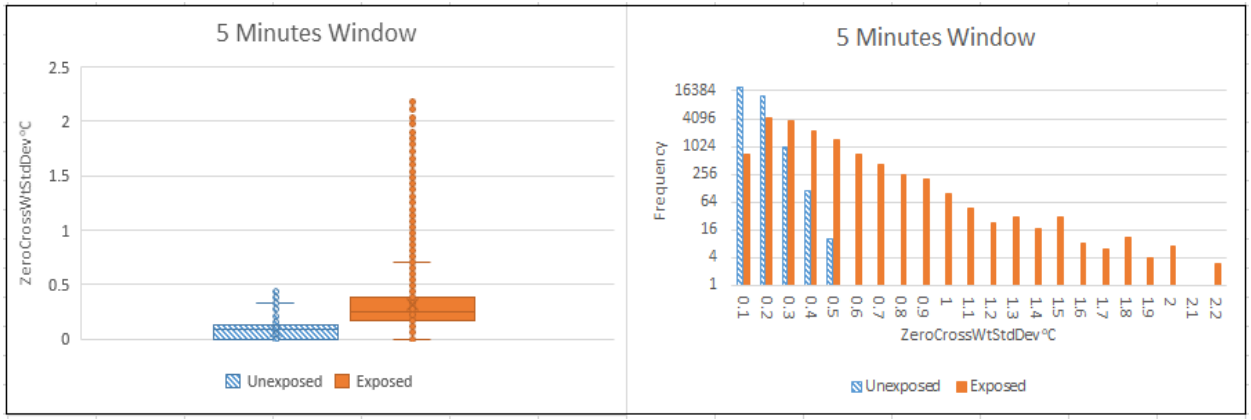

Figure 15: Class distribution for ZeroCrossWtStdDev feature in 5 Minutes Window dataset

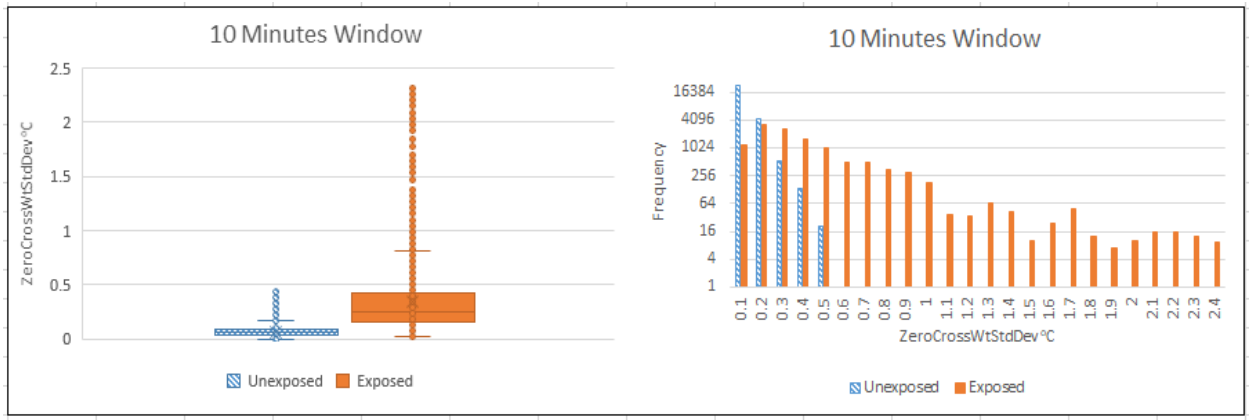

Figure 16: Class distribution for ZeroCrossWtStdDev feature in 10 Minutes Window dataset

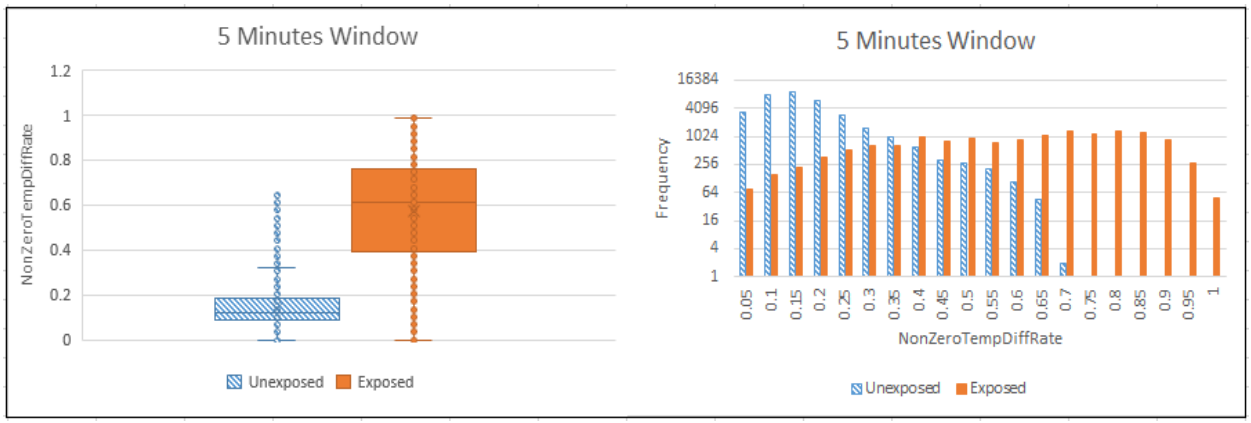

Figure 17: Class distribution for NonZeroTempDiffRate feature in 5 Minutes Window dataset

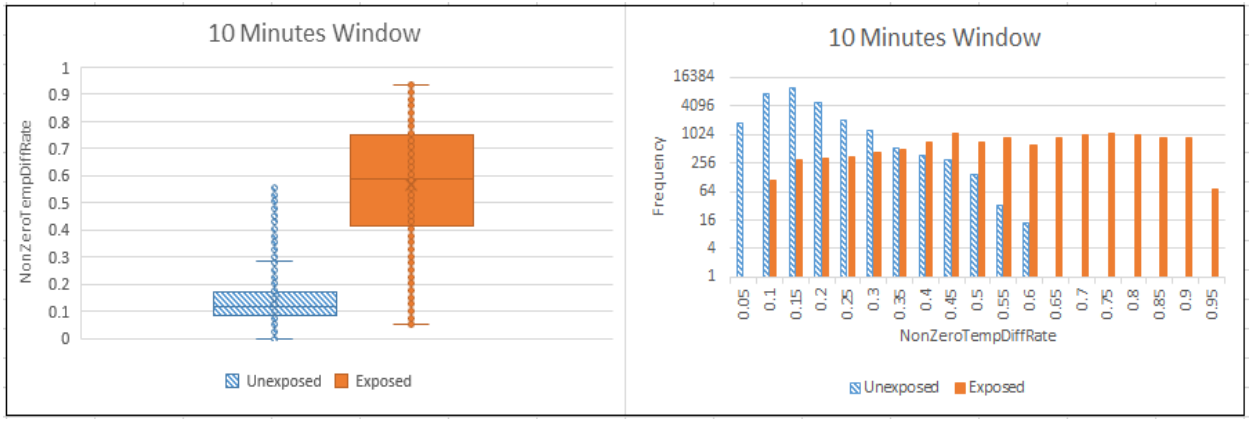

Figure 18: Class distribution for NonZeroTempDiffRate feature in 10 Minutes Window dataset

Figure 11 to Figure 18 gives us an overview of individual feature based class distribution for both for 5 minutes and 10 minutes window datasets. As can be observed in the histograms for individual features, unexposed data points are right skewed and exposed data points left skewed in the dataset; and they overlap in the middle. By observing box plots, we can state that data points present between lower quartile and upper quartile, for both unexposed and exposed, do not overlap with each other across all the features; making data linearly separable. Data points present between lower quartile to lowest observation and upper quartile to highest observation, for both unexposed and exposed, overlap with the data points from other class. Above analysis is based on time series dataset with 5 seconds interval, for a given window size, feature wise class distribution is different for different interval time series.

4.1.2 Class Distribution in Feature Space

This section discusses class distribution in two-dimensional feature space for our dataset. Dataset generated using temperature time series with 5 seconds interval, both with 5 minutes and 10 minutes window, was used and data points were plotted on scatter plot to show class boundaries using StdDevTempDiff, ZeroCrossRate, ZeroCrossWtStdDev and NonZeroTempDiffRate features as axes. Data was not standardized for scatter plot generation.

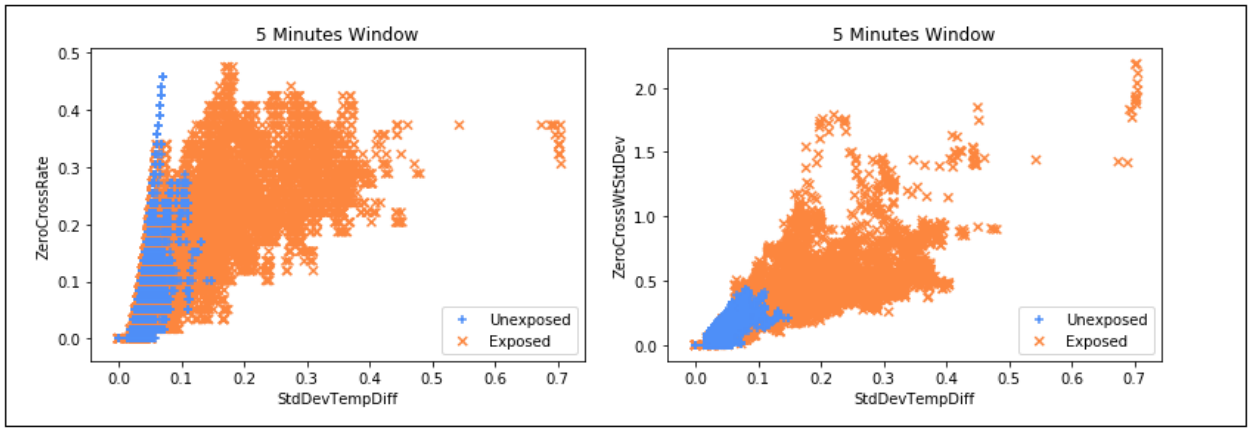

Figure 19: Scatter plot 1 for class boundaries in 5 Minutes Window dataset

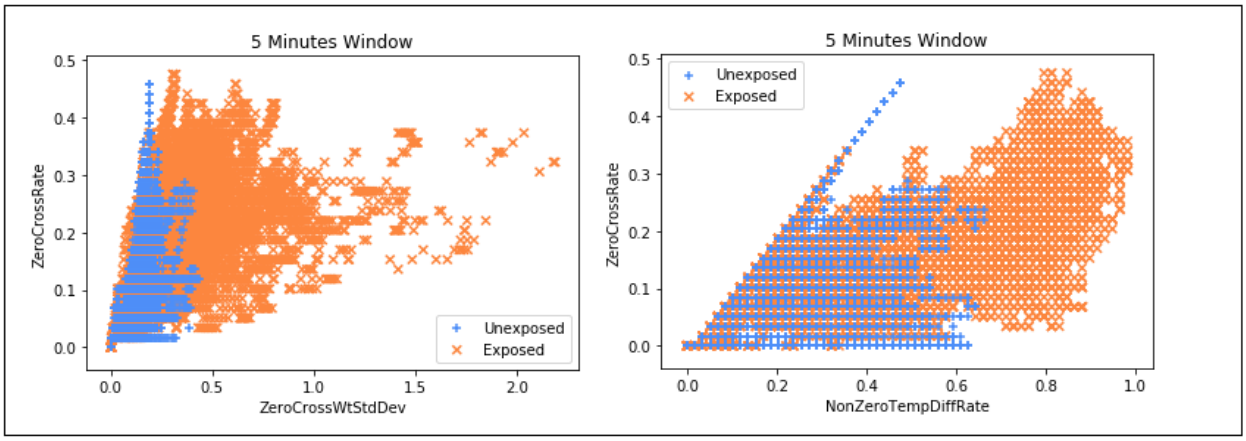

Figure 20: Scatter plot 2 for class boundaries in 5 Minutes Window dataset

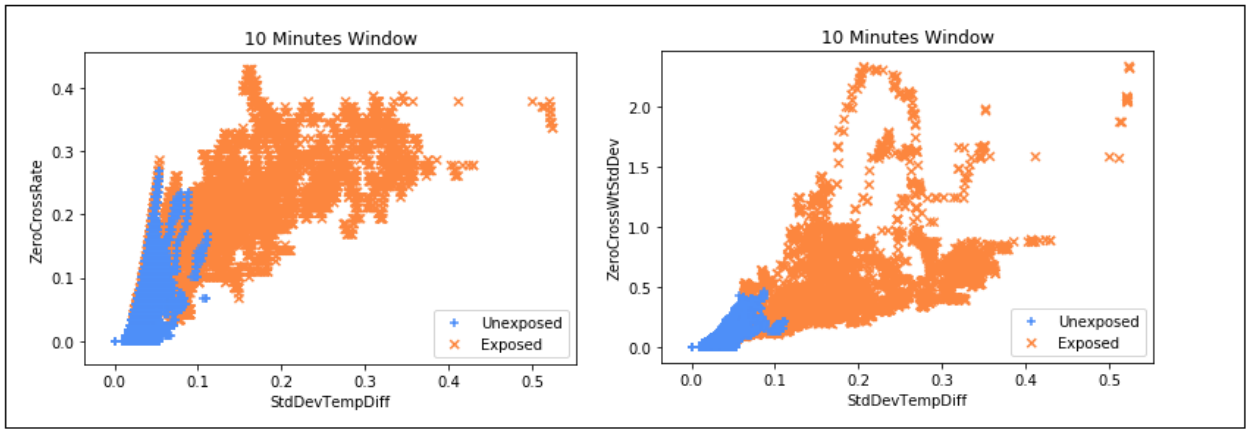

Figure 21: Scatter plot 1 for class boundaries in 10 Minutes Window dataset

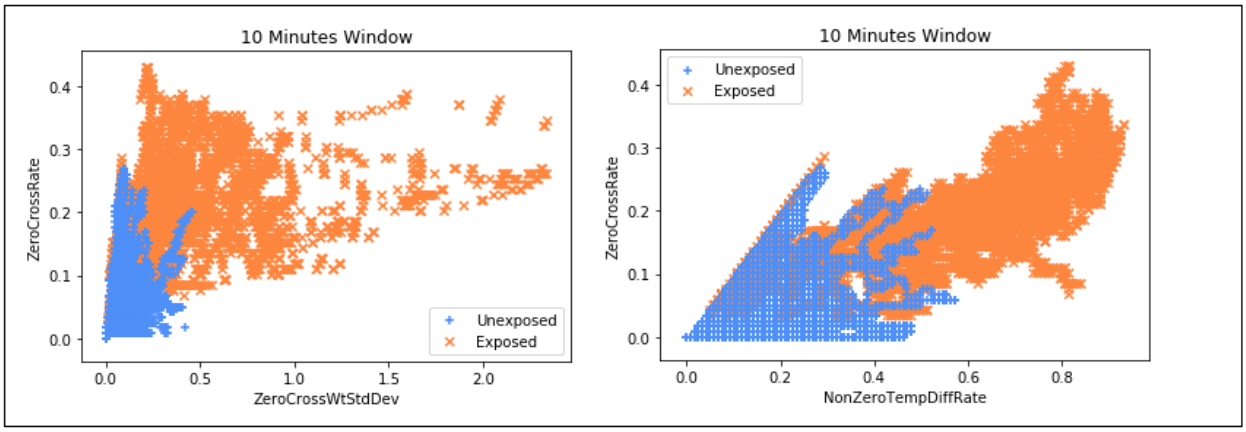

Figure 22: Scatter plot 2 for class boundaries in 10 Minutes Window dataset

As can be seen from Figure 19 to Figure 22, class boundaries overlap with each other in two-dimensional feature space. However, unexposed data points have lower values and are prominently clustered around origin and exposed data points comparatively have higher values and are spread away from the origin.

4.2 Data Correlation

This section discusses data correlation in datasets generated using temperature time series with 5 seconds interval, both with 5 minutes and 10 minutes window. Pandas package [25] for python was used to calculate Pearson correlation coefficients. Data was not standardized for calculating correlation coefficients.

Table 1: Pearson correlation coefficient metrics for 5 minutes window dataset:

	StdDevTemp-Diff	ZeroCross-Rate	ZeroCrossWt-StdDev	NonZeroTemp-DiffRate
StdDevTemp-Diff	1.00	0.74	0.85	0.89
ZeroCross-Rate	0.74	1.00	0.73	0.79
ZeroCrossWt-StdDev	0.85	0.73	1.00	0.79
NonZeroTemp-DiffRate	0.89	0.79	0.79	1.00

Table 2: Pearson correlation coefficient metrics for 10 minutes window dataset:

	StdDevTemp-Diff	ZeroCross-Rate	ZeroCrossWt-StdDev	NonZeroTemp-DiffRate
StdDevTemp-Diff	1.00	0.80	0.80	0.92
ZeroCross-Rate	0.80	1.00	0.63	0.84
ZeroCrossWt-StdDev	0.80	0.63	1.00	0.77
NonZeroTemp-DiffRate	0.92	0.84	0.77	1.00

Table 1 and Table 2 shows Pearson correlation coefficients between feature variables used to train classifiers. Pearson correlation coefficient is used to understand linear relationship between variables, if correlation coefficient is positive then there is positive correlation between variables [27]. If correlation coefficient is negative, then there is negative correlation between variables and if correlation coefficient is zero then there is no correlation between variables [27].

4.3 Performance Metrics

In our approach we use binary classification to identify anomalous subsequences. Sensitivity, specificity and macro F1-score are three performance measures we have used to evaluate performance of SVM, Logistic Regression and Random Forest.

Unexposed temporal temperature subsequences are Positive class instances, and exposed temporal temperature subsequences are defined as normal and form Negative class instances.

Table 3: Confusion Matrix for Anomaly Detection Technique:

	Detection	
Truth	Anomaly (Unexposed Subsequence)	Normal (Exposed Subsequence)
Anomaly (Unexposed Subsequence)	True Positive (TP)	False Negative (FN)
Normal (Exposed Subsequence)	False Positive (FP)	True Negative (TN)

Table 3 shows confusion matrix [10] used to measure performance of binary classifiers.

Sensitivity helps us understand the probability with which the proposed method identifies the unexposed subsequences [10]. Mathematically sensitivity is defined as [10],

Sensitivity = TP/(TP+FN)

Another performance measure used in evaluation is specificity, it gives us idea about how effectively a classifier identifies negative classes, it is defined as [10, 28],

Specificity = TN/(TN+FP)

Precision is the ratio of correctly identified positive class instances by the classifier to total positive class instances in dataset and is defined as [28, 29],

Precision = TP/(TP+FP)

Formula of recall is similar to that of sensitivity and is defined as [28],

Recall = TP/(TP+FN)

Macro/Average F1-score is defined as arithmetic average of F1-score of both positive and negative class [28, 29]. F1-score formula is [29],

F1-score = 2 * (Precision * Recall) / (Precision + Recall)

Performance metrics used for evaluation of system should take into consideration the underlying class distribution in the dataset. Macro F1-score, sensitivity and specificity together help in evaluating classifier performance for both positive class and negative class instances.

4.4 Statistical Analysis of Supervised Learning Models

To demonstrate the effectiveness of the proposed anomaly detection technique for crowdsensed temporal temperature data, we have applied it to the temperature data collected by volunteers using Kestrel DROP Sensors. We labeled the collected data and split it into 40% training sets and 60% testing sets, 5-fold cross-validation was used during training phase and 10 different

test-train sets were used for validation purpose. We used averaged macro F1-score, sensitivity, specificity values for comparing classifier performances.

Parameters used to train SVM, Logistic Regression and Random Forest classifiers from "scikit-learn" package [26] are given below.

Table 4: SVC classifier parameters:

Parameter Name	Value
C	1.0
kernel	linear
degree	3
gamma	auto
coef0	0.0
probability	False
shrinking	True
tol	1e-3
cache_size	200 MB
class_weight	None
verbose	False
max_iter	-1
decision_function_shape	None
random_state	0

Table 5: Logistic Regression classifier parameters:

Parameter Name	Value
penalty	l2
dual	False
C	1.0
fit_intercept	True
intercept_scaling	1
class_weight	None
max_iter	100
random_state	None
solver	liblinear
tol	1e-4
multi_class	ovr
verbose	0
warm_start	False
n_jobs	1

Table 6: Random Forest classifier parameters:

Parameter Name	Value
n_estimators	100
criterion	gini
max_features	auto
max_depth	None
min_samples_split	2
min_samples_leaf	1
min_weight_fraction_leaf	0
max_leaf_nodes	None
min_impurity_split	1e-7
bootstrap	True
oob_score	False
n_jobs	1
random_state	None
verbose	0
warm_start	False
class_weight	None

Table 4 shows SVC classifier parameters we used to train SVM model. The "kernel" parameter allows us to specify the type of kernel to be used to train SVM classifier. Table 5 shows parameters we used to train Logistic Regression classifier. The "penalty" parameter allows us to specify norm penalty to be used and "solver" parameter helps us specify the algorithm to employ in optimization problem. Table 6 shows parameters we used to train Random Forest classifier. The "n_estimators" parameter specifies number of trees to use in Random Forest classifier and "criterion" parameter specifies the criteria to measure the quality of split.

The parameters shown in Table 4 to Table 6 were used to train the classifiers using all four extracted features; and identical parameters were used to train classifiers using only zero-crossing rate feature.

4.4.1 Classifier Performance using All Extracted Features

We trained the SVM, Logistic Regression and Random Forest classifiers with all the four

extracted features and evaluated their performance using sensitivity, specificity and macro F1-score. Classifiers were trained using datasets generated with two different window sizes (5 minutes and 10 minutes window) on time series with 5, 10, 20, 30, 60 seconds intervals. Evaluation metric compares performance measure scores of binary classifiers trained on historical data with different time series intervals.

Table 7: Sensitivity metrics with 5 minutes window and all features:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.96	0.96	0.97	0.96	0.97
Logistic Regression	0.96	0.96	0.96	0.96	0.97
Random Forest	0.93	0.93	0.94	0.94	0.95

Table 8: Sensitivity metrics with 10 minutes window and all features:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.96	0.96	0.97	0.97	0.97
Logistic Regression	0.96	0.96	0.96	0.97	0.96
Random Forest	0.93	0.93	0.94	0.94	0.94

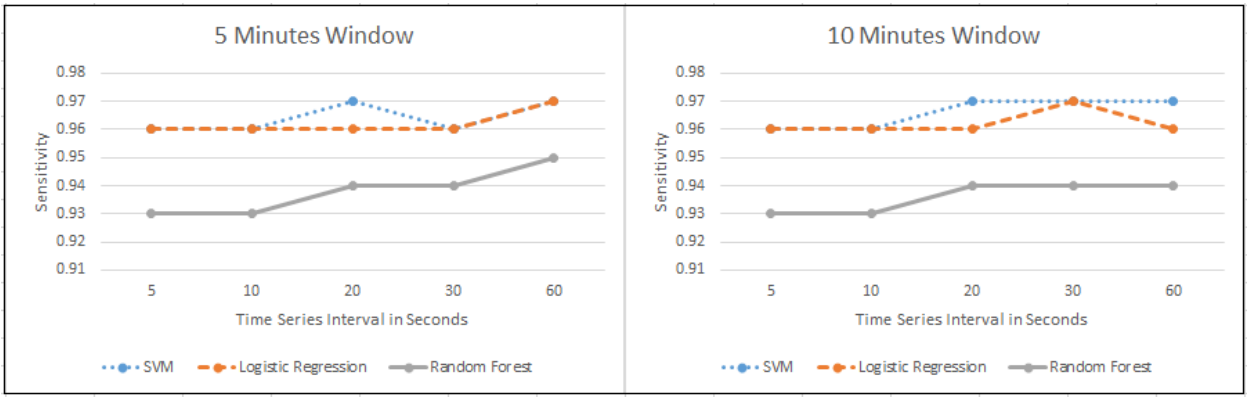

Figure 23: Sensitivity analysis for Models trained on all features

We can see from Figure 23, SVM consistently shows equal or better performance than Logistic

Regression and Random Forest. Sensitivity increases with increase in time interval for all the classifiers for given dataset. In 10 minutes window there are more readings than 5 minutes window. Therefore, as expected SVM shows slightly better performance with 10 minutes window size than with 5 minutes window. Logistic Regression and Random Forest show slight dip in performance at 60 seconds interval with 10 minutes window than with 5 minutes window.

Table 9: Specificity metrics with 5 minutes window and all features:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.80	0.82	0.83	0.82	0.76
Logistic Regression	0.80	0.82	0.83	0.82	0.75
Random Forest	0.82	0.84	0.85	0.85	0.79

Table 10: Specificity metrics with 10 minutes window and all features:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.84	0.83	0.86	0.85	0.80
Logistic Regression	0.84	0.85	0.86	0.85	0.80
Random Forest	0.83	0.84	0.85	0.88	0.81

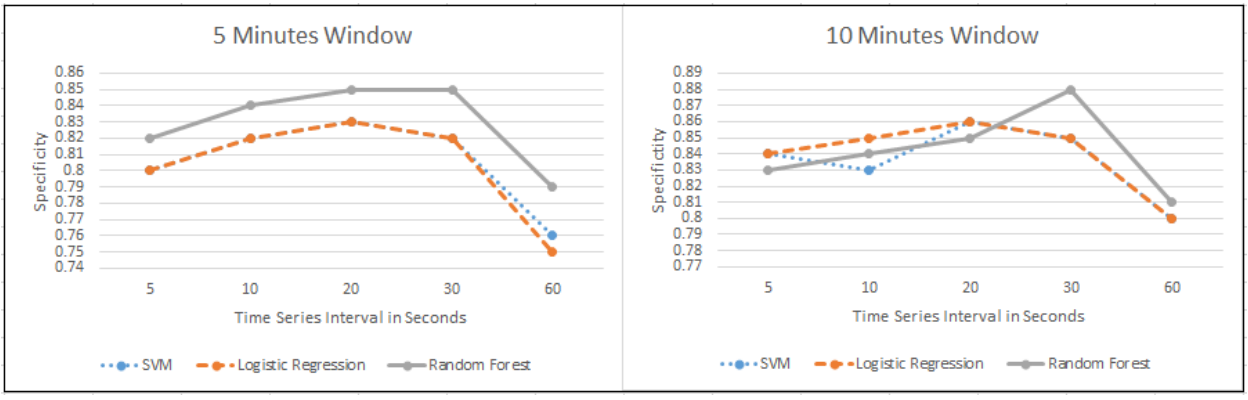

Figure 24: Specificity analysis for Models trained on all features

As can be seen in Figure 24, specificity of all classifiers increases slightly, except for SVM at 10

seconds interval with 10 minutes window, up to 20 seconds time interval and then decreases as expected with increase in time interval. As time interval in temperature time series increases, temperature readings in given window size decreases. Temperature time series with 5 seconds interval, contains 60 temperature readings in 5 minutes window and 120 temperature readings in 10 minutes window. Time series with 60 seconds time interval, contains 5 temperature readings in 5 minutes window and 10 temperature readings in 10 minutes window. This affects feature extraction and results in decreased classifier performance using specificity as measure with increase in time interval. For given time interval, 10 minutes window has more temperature readings than 5 minutes window. Therefore, classifiers performances with 10 minutes window are better than with 5 minutes window. Random Forest performs better than other classifiers with 5 minutes window. Overall, Logistic Regression and SVM perform equally with both 5 minutes and 10 minutes window, except at 60 seconds interval with 5 minutes window and at 10 seconds interval with 10 minutes window.

Table 11: Macro F1-score metrics with 5 minutes window and all features:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.89	0.90	0.91	0.90	0.88
Logistic Regression	0.89	0.90	0.91	0.90	0.88
Random Forest	0.88	0.89	0.90	0.90	0.87

Table 12: Macro F1-score metrics with 10 minutes window and all features:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.91	0.90	0.92	0.92	0.90
Logistic Regression	0.91	0.91	0.92	0.92	0.89
Random Forest	0.88	0.88	0.90	0.91	0.88

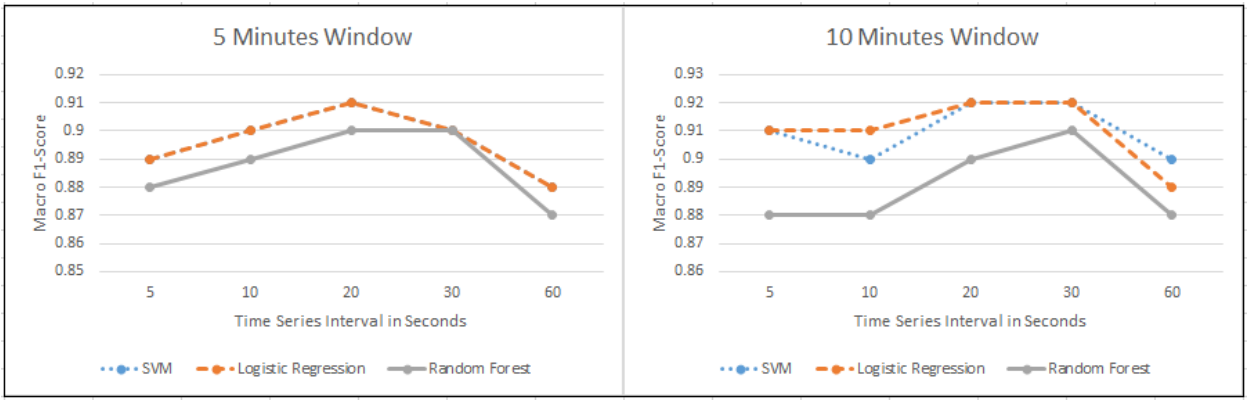

Figure 25: Macro F1-score analysis for Models trained on all features

From Figure 25, we can see that both SVM and Logistic Regression outperform Random Forest with 5 minutes and 10 minutes window. SVM and Logistic Regression both perform equally with 5 minutes window. With 10 minutes window, Logistic Regression slightly performs better at 10 seconds interval than SVM and SVM performs better at 60 seconds interval. Overall, both show similar performances. As expected, both Logistic Regression and SVM perform better with 10 minutes window than with 5 minutes window. Performance of Random Forest dips slightly at 10 seconds interval with 10 minutes window than with 5 minutes window. Readings decrease with increased time interval for given window size, with 30 seconds time interval we have 10 readings with 5 minutes window and 20 readings with 10 minutes window. Therefore, classifier performance decreases after 30 seconds time interval using macro F1-score as measure.

By considering performance measures sensitivity, specificity and macro F1-score, we see SVM and Logistic Regression perform consistently well. As seen in Figure 24, Random Forest performs better with 5 minutes window using specificity as measure but SVM and Logistic Regression perform better using sensitivity and macro F1-score measures. Analyzing macro F1-score and specificity performance metrics we have observed that both SVM and Logistic Regression overall show equal performance but SVM performs better using sensitivity measure.

Also, with 10 minutes window, performance of SVM and Logistic Regression is same at 20, 30 and 60 seconds time interval considering specificity as measure but SVM improves its performance using sensitivity as measure. Overall, SVM shows better and more consistent performance than Logistic Regression and Random Forest. SVM classifier with 10 minutes window performs better than one using 5 minutes window. Also, it is observed that overall classifier performance decreases with increase in time interval after 20 seconds interval and as expected at 60 seconds interval classifier performance is at lowest.

4.4.2 Classifier Performance using Zero-Crossing Rate

To test performance of a lightweight classifier, we trained SVM, Logistic Regression and Random Forest classifiers using only zero-crossing rate feature. Zero-crossing rate feature effectively captures fluctuations observed in temperature sensors readings exposed to ambient atmosphere. As per our empirical analysis this is an important aspect separating exposed and unexposed temperature sensor readings. An efficient lightweight model for anomaly detection will improve computational efficiency and provide opportunity to extend the approach to low computational power devices.

Trained supervised learning models were evaluated using sensitivity, specificity and macro F1-score measures. Different training models were created with 5 minutes and 10 minutes window sizes for temperature time series with 5, 10, 20, 30 and 60 seconds intervals.

Table 13: Sensitivity metrics with 5 minutes window and Zero-Crossing Rate feature:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.92	0.93	0.93	0.91	0.88
Logistic Regression	0.93	0.93	0.93	0.94	0.92
Random Forest	0.91	0.91	0.91	0.90	0.86

Table 14: Sensitivity metrics with 10 minutes window and Zero-Crossing Rate feature:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.94	0.94	0.94	0.94	0.93
Logistic Regression	0.94	0.94	0.94	0.95	0.95
Random Forest	0.92	0.95	0.93	0.92	0.95

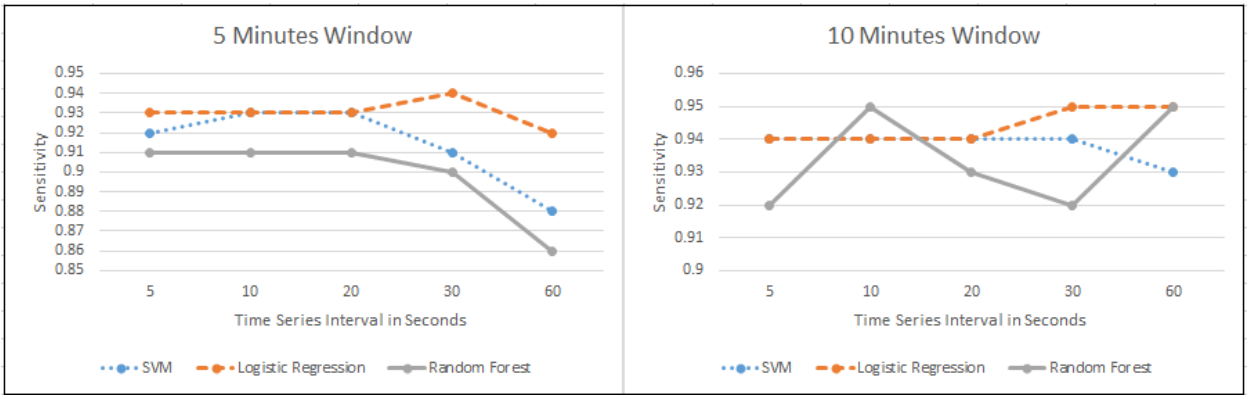

Figure 26: Sensitivity analysis for Models trained on Zero-Crossing Rate feature

As can be seen in Figure 26, using sensitivity as measure Logistic Regression performs better than other classifiers. SVM performance decreases from 30 seconds interval compared to Logistic Regression. Logistic Regression and SVM both perform better than Random Forest. Random Forest performance with 10 minutes window is not consistent across the different time series intervals. In Random Forest algorithm, a classification tree is built by recursively dividing the data into homogeneous regions based on class values [19]. Random Forest uses multiple classification trees to determine the class of a data point and may not show consistent performance across different time intervals based on underlying data distribution. Therefore, in case of only using zero crossing rate feature, sensitivity gain or loss for Random Forest classifier may come at cost of specificity score at corresponding time interval, resulting in inconsistent behavior. We can observe this behavior in Figure 26 and in Figure 27 at 30 seconds and 60

seconds time intervals with 10 minutes window.

All the classifiers perform better with 10 minutes window than with 5 minutes window.

Table 15: Specificity metrics with 5 minutes window and Zero-Crossing Rate feature:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.69	0.75	0.76	0.77	0.70
Logistic Regression	0.65	0.73	0.74	0.69	0.59
Random Forest	0.70	0.80	0.81	0.82	0.77

Table 16: Specificity metrics with 10 minutes window and Zero-Crossing Rate feature:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.75	0.81	0.80	0.82	0.76
Logistic Regression	0.74	0.80	0.79	0.80	0.73
Random Forest	0.75	0.79	0.81	0.86	0.73

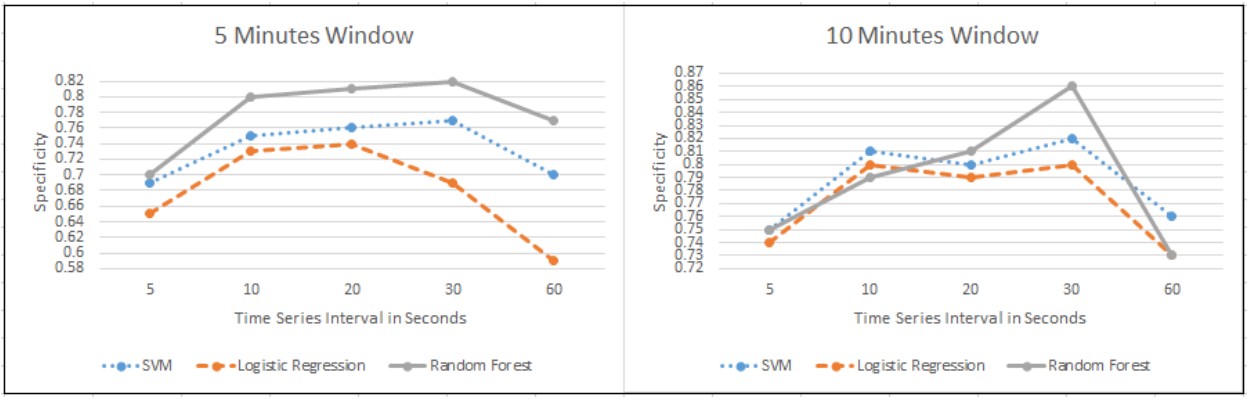

Figure 27: Specificity analysis for Models trained on Zero-Crossing Rate feature

From Figure 27, comparing classifiers using specificity as measure, we can see that Random Forest performs better than SVM and Logistic Regression with 5 minutes window but with 10 minutes window performance is not consistent across different time intervals. SVM performs consistently better than Logistic Regression with both 5 minutes and 10 minutes window. All

classifiers perform better with 10 minutes window than with 5 minutes window. The performance of classifiers decreases at 60 seconds time interval as temperature observations decrease with increase in time interval for given window size.

Table 17: Macro F1-score metrics with 5 minutes window and Zero-Crossing Rate feature:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.81	0.85	0.85	0.84	0.79
Logistic Regression	0.81	0.84	0.84	0.83	0.76
Random Forest	0.82	0.86	0.86	0.85	0.81

Table 18: Macro F1-score metrics with 10 minutes window and Zero-Crossing Rate feature:

	Temperature Time Series Intervals (Seconds)				
Supervised Learning Models	5	10	20	30	60
SVM	0.85	0.88	0.87	0.88	0.85
Logistic Regression	0.85	0.88	0.87	0.88	0.85
Random Forest	0.84	0.88	0.87	0.89	0.85

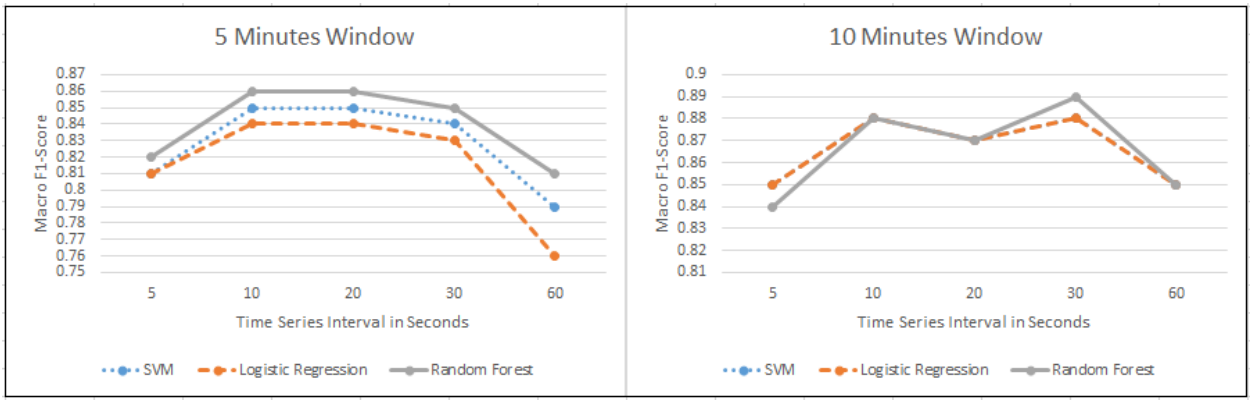

Figure 28: Macro F1-score analysis for Models trained on Zero-Crossing Rate feature

Using macro F1-score as measure, Random Forest performs better than other classifiers with 5 minutes window. All classifiers perform equally well with 10 minutes window, except at 5 and 30 seconds intervals. SVM performs better than Logistic Regression with 5 minutes window and

their performance is identical with 10 minutes window. For given time interval, 10 minutes window has more temperature readings than 5 minutes window. Therefore, all classifiers perform better with 10 minutes window than with 5 minutes window.

Comparing overall performance of all the classifiers using sensitivity, specificity and macro F1-score, we notice that performance of SVM is better than Random Forest and Logistic Regression. SVM performs better than Logistic Regression using specificity and macro F1-score measures. Sensitivity gains for Logistic Regression against SVM comes at the cost of its decreased specificity scores. Random Forest performs better than SVM with 5 minutes window using macro F1-score and specificity as measure, but its performance is not consistent with 10 minutes window. SVM performs better than Random Forest using sensitivity as measure. Overall, SVM model with 10 minutes window performs better and more consistently than other models. Temperature time series subsequence has less number of temperature readings with increase in time interval for given window size. Therefore, as expected SVM classifier performance decreases with increase in time interval and the performance is lowest at 60 seconds time interval.

4.4.3 SVM Classifier Performance Comparison

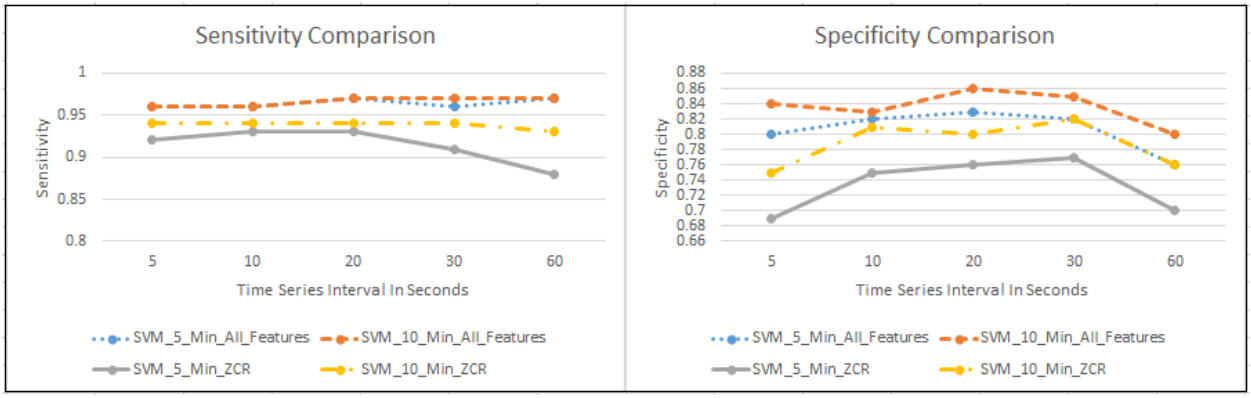

Figure 29: SVM Classifier comparison using Sensitivity and Specificity measures

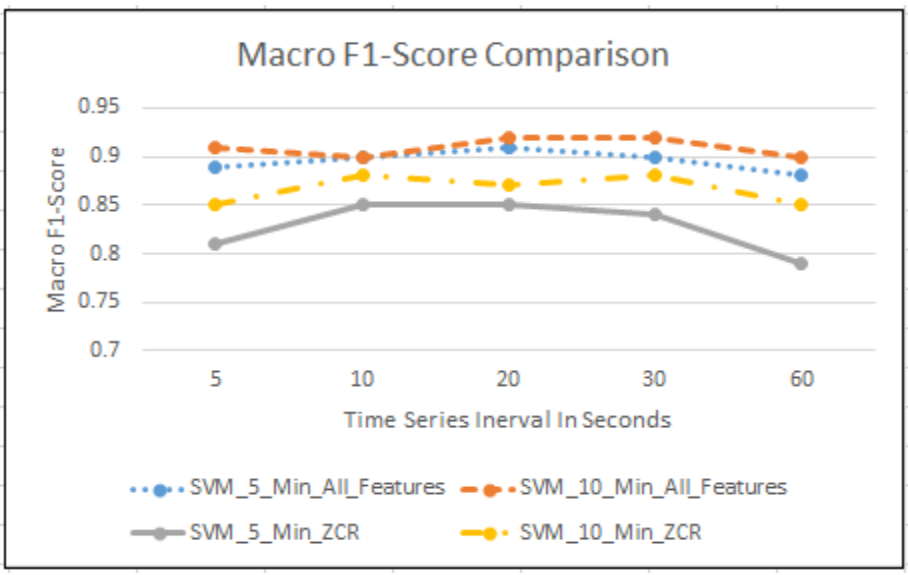

Figure 30: SVM Classifier comparison using Macro F1-score

As can be seen in Figure 29 and in Figure 30, SVM classifier gives consistent results on sensitivity, specificity and macro F1-score measures. Using all the three measures and comparing classifiers trained with same features, we can see classifier performance with 10 minutes window is higher than classifier performance with 5 minutes window. Comparatively, SVM classifiers trained with only zero-crossing rate feature show lower performance using all performance measures. In comparison with SVM classifiers trained with all features, SVM classifiers trained with zero-crossing rate using sensitivity as measure show relatively small performance decrease than macro F1-score and specificity. Specifically using specificity as measure SVM classifier performance decreases sharply when only trained with zero-crossing rate feature. Also, from Figure 23 and 24 we can observe that SVM classifiers trained with 5 minutes window size with all the four features show better performance when compared with SVM classifiers trained with 10 minutes window size and only zero-crossing rate feature. This suggests better or equal performance can be achieved with smaller window size by increasing total number of features.

CHAPTER 5

RELATED WORK

5.1 Outlier Detection in Temporal Temperature Data

[30] Ma et al. proposes an algorithm to detect outliers based on sliding window in temperature time series in meteorological sensor network. As discussed in the paper meteorological observations forms the basis of meteorology and atmospheric science; and the data is used in fields like agriculture, forestry, traffic, hydrology and health. They have used temperature data collected from weather stations and meteorological sensor networks for the experimental analysis. The low cost meteorological sensors are prone to outside factors, which leads to certain errors getting introduced in the data [30]. To improve the accuracy of the data collected using meteorological sensors, the study proposes a model to detect the outliers in the temperature time series and correct the values. They used an Autoregressive Prediction model to do the predictions based on historical time series data. The study uses data points in a given sliding window as an input parameter, and then uses this information to predict future value. The observed value and predicted value are compared, and if the observed value lies outside certain threshold then the value is defined as an outlier. The outlier detection algorithm used in the study is based on assumption that next temperature value in the time series can be derived from the historical temperature values in a given sliding window. The study claims to improve the performance of outlier detection and its correction.

5.2 Outlier Detection Techniques

Multiple factors like domain, type of outlier, real time or batch processing system, and

availability of labeled data play an important role in applying an outlier detection technique to the problem. Visualization gives insights for identifying underlying patterns, distributions in the data and helps in defining outlier patterns. In this section we will discuss certain outlier detection techniques and their use cases.

5.2.1 3-Sigma Method

In this method, from the given dataset mean and standard deviations is calculated and then if a data point is away from the mean by certain number of standard deviations, default value is 3 times, then that data point is considered as an outlier [31]. As outliers are part of the mean calculation in this method, it may affect the mean calculation to a certain extent if outliers form considerable percentage of the dataset [31]; and it may become difficult to detect the outliers. Also, if the data is a series-based data like time series, or spatial-temporal data where data points have spatial relation then this method does not consider such relations between data points, and so may become ineffective.

5.2.2 Outlier Detection using K-Means Clustering

Non-supervised clustering methods like k-means clustering are useful if the labeled training dataset is not available. In these methods, we can have multiple features as dimensions of a data point. These dimensions are then used to cluster the nearest data points together by using methods like Euclidean distance calculation [32]. Based on user specified parameter k, total k clusters are formed, and each cluster has a centroid associated with it. We must define what an outlier means in our system. Then we can use those dimensions of centroid in defining the outliers for the given system [32]. The computational complexity $O(n^2m)$ is defined by number of dimensions (m) and data points (n) [32]. Therefore, the method is not suitable for high dimensional datasets. Modified distance-based methods were proposed to improve the efficiency,

where a ranking is generated based on the distance of a point from its k^{th} nearest neighbor and top n points are declared as outliers [33]. A partition-based algorithm is also proposed which partitions input data points into disjoint datasets to improve the computational efficiency [33].

5.2.3 Neural Networks

These are non-parametric, and model-based approaches and they generalize to unknown data and can learn complex class boundaries [32]. Once trained they act as classifiers. We must train neural networks multiple times and traverse the entire dataset to settle the model [32]. They identify the patterns and focus on important dimensions, but dimension reduction using feature selection is useful [32].

- Supervised Neural Networks: Classified data is used to train network and class is used to adjust the weight and threshold, such that the neural network can classify the input correctly [32]. Techniques like Multilayer Perceptron (MLP) interpolate well but perform poorly for extrapolation. Therefore, cannot classify the unseen instances outside the bounds of training set [32]. This aspect was used for fault identification in aircraft engine vibration signature and to monitor oil pipeline flows [32].
- Unsupervised Neural Networks: Supervised neural net cannot be used when we do not have labeled data set. In unsupervised neural nets, autonomous clustering of input vector takes place to model the data distribution and to define normal, abnormal classes [32]. They too need training data set to learn. They rely on identifying the common features in input vectors and their values, to topologically model the data distribution [32]. Grow when required (GWR) evolutionary neural network can adjust and model dynamic data distribution [32]. GWR is used in mobile inspection robot, online learning, and novelty detection [32].

5.3 Outliers in Temporal Data

In time series outlier analysis, anomalies are identified in the behavior attribute of the data point where context attribute is time [34]. Domain plays an important role in determining patterns for outlier. In temporal outlier detection, temporal aspect is important and abnormal changes, subsequences and temporal patterns in the series are used to define outliers [34].

5.3.1 Outliers in Time Series

- Outlier Detection in Time Series Database: Given a time series database, outlier time series are detected either by calculating outlier scores of entire time series or using windowing, outliers scores are calculated for the window and then aggregated to define outlier score of the time series [34]. In this case either entire time series or subsequence of the time series is identified as an outlier. Unsupervised methods like k-means clustering and supervised methods like SVM are used to identify outlier time series [34].
- Outlier Detection in a Given Time Series: Given a time series, point outliers or a subsequence is identified as an outlier [34]. Prediction models and Profile based models are used to identify the point outliers. Prediction Models like Multilayer Perceptron (MLP), Autoregressive Integrated Moving Average (ARIMA) are used to predict a point outlier. Outlier score for the point is calculated by its deviation from the predicted value and then it is identified as an outlier [34]. In Profile Models a normal profile is maintained, and point is identified as an outlier based on its deviation from the normal profile [34]. For example, in case of OS multivariate performance metric time series, a normal profile and variance vector is maintained, and outlier is determined by calculating deviation of a data point from both [34]. To identify a subsequence as an outlier subsequence in a time series, a given subsequence is compared with all its nearest non-

overlapping subsequences and an outlier subsequence is determined based on the largest distance [34].

5.3.2 Outliers in Data Stream

- Evolving Prediction Models: Given a data stream, outliers are identified based on model which is updated as new data arrives, which helps in capturing normal trends in the data [34]. An online discounting learning algorithm, dynamically maintained clusters are examples of the type [34].
- Distance Based Outliers for Sliding Window: Given a stream of data, outliers in a window are identified based on the distance with other points, in local or global context [34]. Algorithm using Indexed Stream Buffer data structure, dynamically maintained clusters are used to identify distance based global outliers [34]. Local Outlier Factor (LOF) algorithm which uses distance between neighbors and given data point to detect outliers is used for distance based local outliers [34].
- Outlier Detection in High Dimensional Stream: Stream Projected Outlier Detector (SPOT) is used to detect outliers in high dimensional data streams [34]. It uses a window-based technique to capture the statistics and is capable of online self-evolution [34].

CHAPTER 6

CONCLUSIONS

Understanding dynamic nature of Urban Heat Island (UHI) phenomenon and corresponding heat exposure effects on individuals and small communities, helps in avoiding health hazards. Crowdsensed temperature data, helps us understand effects of local heat-generating activities introducing dynamicity to the UHI effect. However, erroneous temperature sensor placement in the controlled environments during the crowdsensing experiments will generate anomalous temporal temperature subsequences affecting the data integrity. This thesis based on empirical observations, temperature readings of a sensor exposed to the ambient environment shows more frequent fluctuations compared to temperature readings of a sensor placed in the controlled environment, proposes a novel approach to detect anomalous sensor placement and filter related temperature subsequences in an almost real-time manner from temperature data stream.

This study uses statistical features like zero-crossing rate based on empirical analysis of patterns observed in temperature time series data. The sliding window approach is used to extract statistical features and classify temporal temperature subsequences using supervised binary classifiers. To validate the effectiveness of our approach we did comparative analysis of Random Forest, SVM and Logistic Regression binary classifiers using both all and reduced feature set. We tested the performance of classifiers using time series with different intervals and two sliding window sizes and noticed correlation between time series interval, window size, feature set and performance. Our system has been able to identify anomalous subsequences effectively, and SVM performed consistently well compared to Logistic Regression and Random Forest.

REFERENCES

1. Ganti, R. K., Ye, F., & Lei, H. (2011). Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine*, *49*(11).

2. Luber, G., & McGeehin, M. (2008). Climate change and extreme heat events. *American journal of preventive medicine*, *35*(5), 429-435.

3. Bourgeois, W., Romain, A. C., Nicolas, J., & Stuetz, R. M. (2003). The use of sensor arrays for environmental monitoring: interests and limitations. *Journal of Environmental Monitoring*, *5*(6), 852-860.

4. Tsou, J., Zhuang, J., Li, Y., & Zhang, Y. (2017). Urban heat island assessment using the Landsat 8 data: a case study in Shenzhen and Hong Kong. *Urban Science*, *1*(1), 10.

5. George, L. A., & Becker, W. G. (2003). Investigating the urban heat island effect with a collaborative inquiry project. *Journal of Geoscience Education*, *51*(2), 237-243.

6. Stewart, I. D. (2011). A systematic review and scientific critique of methodology in modern urban heat island literature. *International Journal of Climatology*, *31*(2), 200-217.

7. Chapman, L., Bell, C., & Bell, S. (2017). Can the crowdsourcing data paradigm take atmospheric science to a new level? A case study of the urban heat island of London quantified using Netatmo weather stations. *International Journal of Climatology*, *37*(9), 3597-3605.

8. Kuras, E. R., Richardson, M. B., Calkins, M. M., Ebi, K. L., Hess, J. J., Kintziger, K. W., ... & Uejio, C. K. (2017). Opportunities and challenges for personal heat exposure research.

9. Singh, K., & Upadhyaya, S. (2012). Outlier detection: applications and techniques. *International Journal of Computer Science Issues*, *9*(1), 307-323.

10. Yu, Y., Zhu, Y., Li, S., & Wan, D. (2014). Time series outlier detection based on sliding window prediction. *Mathematical Problems in Engineering*, *2014*.

11. Dietterich, T. G. (2002, August). Machine learning for sequential data: A review. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)* (pp. 15-30). Springer, Berlin, Heidelberg.

12. Gouyon, F., Pachet, F., & Delerue, O. (2000, December). On the use of zero-crossing rate for an application of classification of percussive sounds. In *Proceedings of the COST G-6 conference on Digital Audio Effects (DAFX-00), Verona, Italy*.

13. Geng, S., Ma, W., Hu, A., Li, W., Guo, D., Lu, J., & Shen, J. (2010, September). Research of zero-crossing detection technology in linear induction motor. In *Electrical Machines (ICEM), 2010 XIX International Conference on* (pp. 1-5). IEEE.

14. Drucker, H., Wu, D., & Vapnik, V. N. (1999). Support vector machines for spam categorization. *IEEE Transactions on Neural networks*, *10*(5), 1048-1054.

15. Widodo, A., & Yang, B. S. (2007). Support vector machine in machine condition monitoring and fault diagnosis. *Mechanical systems and signal processing*, *21*(6), 2560-2574.

16. Min, J. H., & Lee, Y. C. (2005). Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters. *Expert systems with applications*, *28*(4), 603-614.

17. Peng, C. Y. J., Lee, K. L., & Ingersoll, G. M. (2002). An introduction to logistic regression analysis and reporting. *The journal of educational research*, *96*(1), 3-14.

18. Subasi, A., & Ercelebi, E. (2005). Classification of EEG signals using neural network and logistic regression. *Computer methods and programs in biomedicine*, *78*(2), 87-99.

19. Cutler, D. R., Edwards, T. C., Beard, K. H., Cutler, A., Hess, K. T., Gibson, J., & Lawler, J. J. (2007). Random forests for classification in ecology. *Ecology*, *88*(11), 2783-2792.

20. Kestrel DROP Certificate of Conformity: https://cdn.shopify.com/s/files/1/0084/9012/files/DROP_Certificate_of_Calibaration.pdf?14953145177891566479

21. Erell, E., Leal, V., & Maldonado, E. (2005). Measurement of air temperature in the presence of a large radiant flux: an assessment of passively ventilated thermometer screens. *Boundary-layer meteorology*, *114*(1), 205-231.

22. Holden, Z. A., Klene, A. E., Keefe, R. F., & Moisen, G. G. (2013). Design and evaluation of an inexpensive radiation shield for monitoring surface air temperatures. *Agricultural and forest meteorology*, *180*, 281-286.

23. Yang, J., Liu, Q., Dai, W., & Ding, R. (2017). Fluid dynamic analysis and experimental study of a low radiation error temperature sensor. *Physics Letters A*, *381*(4), 177-183.

24. Barde, M. P., & Barde, P. J. (2012). What to use to express the variability of data: Standard deviation or standard error of mean?. *Perspectives in clinical research*, *3*(3), 113.

25. pandas: http://pandas.pydata.org/pandas-docs/stable/

26. scikit-learn: http://scikit-learn.org/stable/

27. Mukaka, M. M. (2012). A guide to appropriate use of correlation coefficient in medical research. Malawi Medical Journal, 24(3), 69-71.

28. Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, *45*(4), 427-437.

29. Yang, Y., & Liu, X. (1999, August). A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 42-49). ACM.

30. Ma, L., Gu, X., & Wang, B. (2017). Correction of Outliers in Temperature Time Series Based on Sliding Window Prediction in Meteorological Sensor Network. *Information*, *8*(2), 60.

31. Posio, J., Leiviskä, K., Ruuska, J., & Ruha, P. (2008). Outlier Detection for 2D Temperature Data. *IFAC Proceedings Volumes*, *41*(2), 1958-1963.

32. Hodge, V., & Austin, J. (2004). A survey of outlier detection methodologies. *Artificial intelligence review*, *22*(2), 85-126.

33. Ramaswamy, S., Rastogi, R., & Shim, K. (2000, May). Efficient algorithms for mining outliers from large data sets. In *ACM Sigmod Record* (Vol. 29, No. 2, pp. 427-438). ACM.

34. Gupta, M., Gao, J., Aggarwal, C. C., & Han, J. (2014). Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, *26*(9), 2250-2267.