Curricular Practical Training Project

Final Report

Utilizing Machine Learning Algorithms to Create Adaptive Artificial Intelligence for Context-Aware Smart Home Monitoring Systems

Palimar A. Rao
57872775
SEAS 2016
Advisor: Dr. Camillo J. Taylor

1. Abstract

This project serves to explore various techniques through which machine learning can be employed to create a smarter home governed by dynamic logic. Two different machine learning algorithms, the supervised learning K-Nearest Neighbour Algorithm and the unsupervised learning K-Means Algorithm are explored. The data for training sets and test-suites originate from the Comcast Beta Network, a collection of Comcast Xfinity set-top boxes and various sensors that collect, anonymize and store real world data in Oracle databases. The smart home control system is fully implemented in software and runs as a Storm topology, which is network of objects called spouts that push data into other objects known as bolts, where computation of the data takes place. The spouts simulate sensors as they push sensor data into the bolt, where they are compiled into training sets and test suites for the machine learning methods. The project results show how such a smart home system can be implemented, and also the greater accuracy of the K-Means algorithm in classifying data as good or bad, as compared to the K-NN algorithm. These results show that it is possible to implement a rudimentary smart home system in software, as well as the greater desirability of using the K-Means algorithm in the smart home control system as opposed to the K-NN algorithm.

2. Introduction

The two primary objectives of this project were to a) create a software abstraction that closely models a smart home control system and b) explore the various methods through which machine learning can be employed to create an artificial intelligence that drives the smart home control system. Such a research project was undertaken because with the prevalence of ubiquitous computing, the necessary hardware to create a smart home systems exists in almost every home. However, the software required to collect data from and control this hardware defined the effectiveness of the smart home system. This project therefore tries to answer the question as to how can a home be made smarter with various kinds of machine learning methods, which is an interesting question by itself.

The smart home control system was modeled based on the Comcast Beta Network. This real world data was then used as training sets for the machine learning algorithms. These machine learning algorithms were not implemented in-house. Rather, JavaML (java Machine-Learning), a collection of machine learning algorithms for data mining tasks, was modified to satisfy unique needs and then trained with the data set from the Comcast Beta Network. The accuracy of these machine learning techniques in predicting the choices of the users was then calculated to rate and rank their effectiveness for use in a smart home control system.

Furthermore, in the process of fulfilling the two objectives stated above, this project explored the various real world applications of big data. Various scenarios were simulated with the real-world data. An example is as follows. A smart home monitoring system was created that continuously gathered data from the user and stored it in a central database. This data was then used by various machine learning algorithms to change the current smart home monitoring system logic in order to better suit the day-to-day preferences of the user. For example, various data such as preferred temperature, light brightness and humidity for a certain user account was stored in the central database. Then this information was used by the machine learning algorithms to change the smart home monitoring system logic to attain optimal climate control that maximizes the comfort, health and safety of that user. Other scenarios involving various aspects of the individual’s life inside the smart home such as entertainment, sanitation and nutrition being controlled by the adaptive smart home monitoring system were also modeled and studied.

3. Implementation of Project Design

The project design can be divided into three major sections. The first section is the data delivery system that collects data from various sensors via Comcast Xfinity set-top boxes and stores it in an Oracle database. The second section is the software modeling of the smart home control system without the logic. The third section is the logic that drives the smart home control system.

3.1. Data Delivery System

From the Philadelphia Metropolitan Area, 69132 Comcast accounts were selected at random. The homes associated with these accounts acted as data sources for the Comcast Beta Network, which is a collection of Comcast Xfinity set-top boxes and various sensors that collect, anonymize and store real world data in Oracle databases. From these homes, data from three sensors: lighting, thermostat and door lock sensor were collected over a period of four weeks. In addition to this, alarm messages emitted by the home security system, as well as security camera access events, not including images, were collected. This data was then transmitted to a single, centrally located Oracle database server via Comcast Xfinity set-top boxes. All sent data was parsed and all senistive information was randomized for anonymity.

3.2. Software Modeling

Since an actual smart home control system could not be produced using hardware, the smart home control system was modeled and tested using software. In order to do this, Storm, a distributed, real-time computation system was used. Storm consists of a network of “bolts” and “spouts” known as a “topology.” A spout object acts a data source that reads in data from a database, and feeds it to the bolt, where computation on the data takes place. The relationship between spouts and bolts is many-to-many. This topology acted as the smart home control system. Therefore, 69132 different topologies were run to act as smart home control systems for all homes from which data was collected.

Since Storm performs stream computation and real-time processing, a playback engine was written in Java to feed data from the Oracle database into the topology with a 10 ms delay. Since all data had timestamps, data was fed into the topology in chronological order. Each of the topologies had 5 spouts to represent the 5 different sensors whose data was being recorded. The playback engine fed data of each sensor to the unique spout that was assigned to that sensor. Once the data was received by the spout, the data would be re-routed to multiple bolts where the actual machine learning algorithm are executed.

3.3. Smart Home Control Logic

The bolts in the topology are where the logic system of the smart home resides. They operate in two modes: training mode and testing mode. In training mode, once the bolts receive the data from the spouts, the data is fed to machine learning methods from JavaML to be used as training sets. In testing mode, the topology makes a prediction of, and compares this prediction with the value of the data that it is to receive. The accuracy of each machine learning method in correctly predicting the value of the data from the sensors is recorded for further analysis.

4. Data Design and Summary

The data used in the training set and test suite for the smart home control system was carefully designed from raw data from the Oracle database. Careful feature selection was carried out on the raw data while constructing the new data model. Such dimension reductionality was carried out because according to the Hughes Phenomenon, the predictive powers of the machine learning algorithms decrease with an increase in the dimensionality of data samples, especially in cases where the training set is of fixed size. This dimension reductionality was carried out in the various bolts of the topology. Therefore the basic data type pushed from spouts to bolts was two dimensional, with attributes for the value of the sensor and a timestamp that records when the value was created. The class type was used to determine the sensors from which the data originated. Such a low dimension proved particularly useful in instance-based learning algorithms, such as the K-nearest neighbour algorithm.

All data types were represented in the topology as Java classes with getter and setter methods. In order to properly represent the relation between the various data types, java classes for accounts, account deployments, premises (physical homes) and zones within premises were also represented by java classes. These classes were related to each other IS-A and HAS-A relationships. The following UML class diagram portrays this relationship.

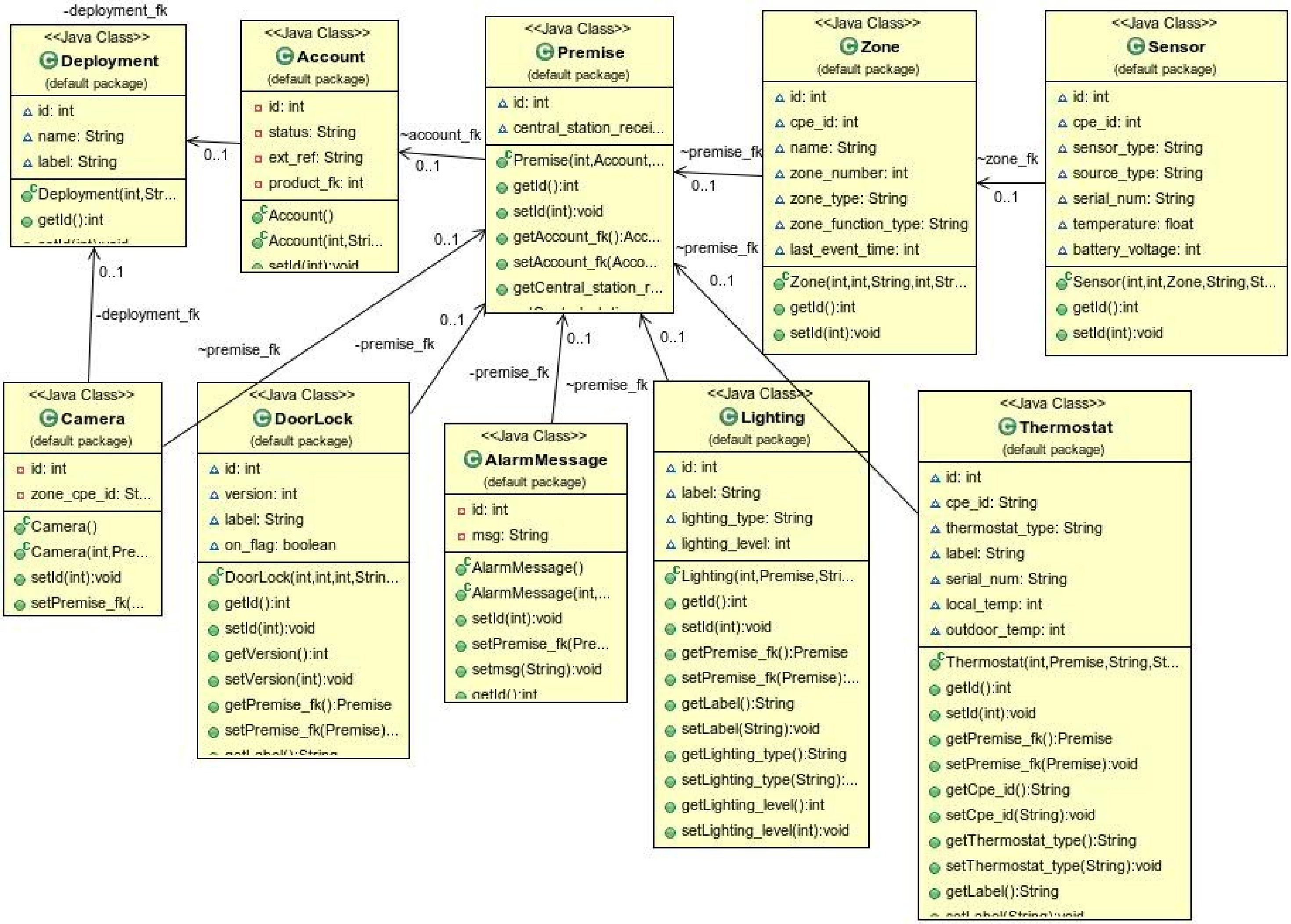

Illustration 1: Relationship between various data types.

5. Data Analysis and Inference

Two different machine learning algorithms were analysed in the smart home control system. The first one is an instance-based learning algorithm known as the K-nearest Neighbours algorithm. The second one is a partitional clustering algorithm known as the K-means algorithm.

5.1. K-Nearest Neighbours Algorithm

The K-Nearest Neighbours algorithm was used because of its simplicity as well as its effectiveness in classifying low dimensional objects. After training the method with a training set, a test suite consisting of real-world data from the Beta network and artificially generated false data was compiled. The machine learning method was then tasked with classifying the data as either correct or erroneous. This is useful in the context of a smart home control system because such learning can detect anomalies in the daily pattern of people living within the home and take corrective measures. Illustration 2 provides a graph that shows how the K-Nearest Neighbours Algorithm performed in separating good from bad data.

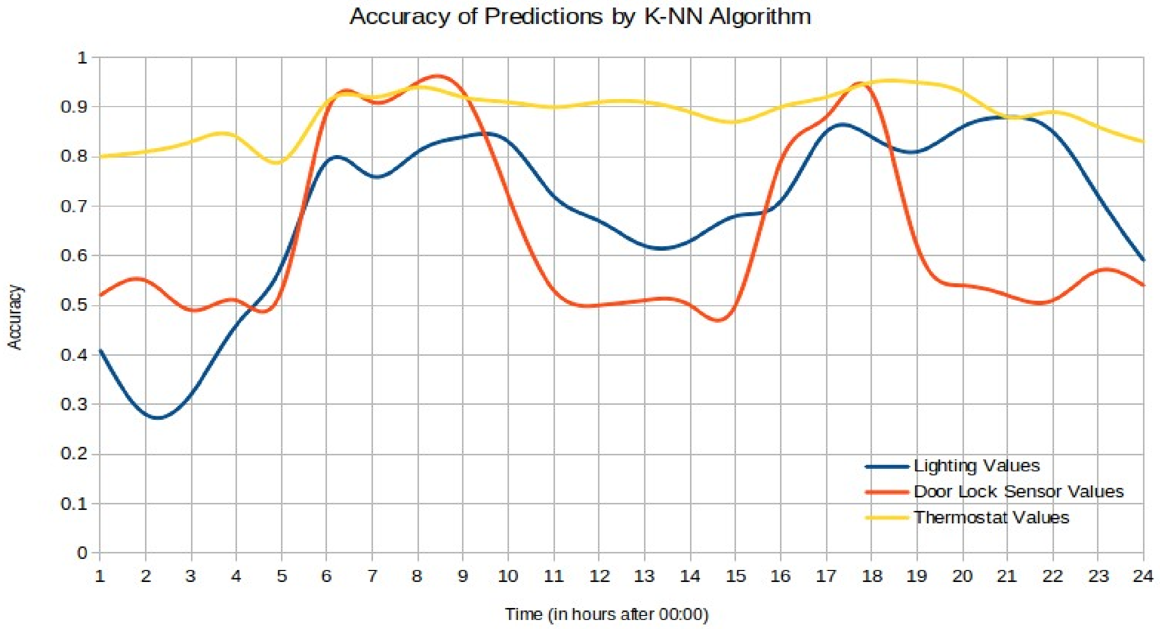

Illustration 2: Accuracy of Predictions by K-NN Algorithm

It can be seen that for all three sensor value predictions, the time periods between 5:00 AM to 9:00 AM and 4:00 PM and 7:00 PM yielded the highest accuracy in recognizing good data from bad data. This is because during these time periods, the sensors recorded most of their data due to the fact that the people living in the houses were most active during this time period. The algorithm was most accurate in predicting thermostat values. This is primarily because of the fact that all data was collected during the summer, and the thermostat value did not change much day-to-day. An almost constant thermostat value resulted in highly accurate predictions by the K-NN Algorithm. It is also interesting to consider the door-lock sensor value predictions. The predictions are accurate only during the time periods between 5:00 AM to 9:00 AM and 4:00 PM and 7:00 PM. This is because the door-lock in homes was used mostly during these times. During the other times, the door-lock sensors emitted a very small number of values. Thus the algorithm did not have too many instances of values from those times to learn from, and there was a 50% chance that the prediction would be true or false, which can be seen in the graph. The predictions for lighting values was also very inaccurate during the times when the the lighting sensors did not emit any values, i.e. during the early hours of the day. However, the fact that lighting values are not binary and range from 0 to 100, as opposed to door-lock sensor values, contributed to the low accuracy of lighting value predictions by the K-NN algorithm.

Overall, 40,560,300 real and generated values from the three sensors were used in the test suite for the K-NN algorithm. The overall accuracy of the algorithm was 83.6%.

5.2. K-Means Algorithm

The K-Means Algorithm divided up the values into two clusters: good data and bad data. Since there is no external classification, the cluster lables are applied after the clusters are created. It was chosen from among other clustering algorithms due to its simplicity. Furthermore, since K is 2, a small value, the algorithm executes in a rapid manner. This allowed the re-execution of the algorithm repeatedly with a different starting mean in order to attain greater accuracy in clustering correct data from false data. Figure 3 provides a graph that shows how the K-Means Algorithm performed in separating good from bad data. Since the algorithm was run multiple times with various means, only the run that achieved the higher predictive accuracy is shown by the graph.

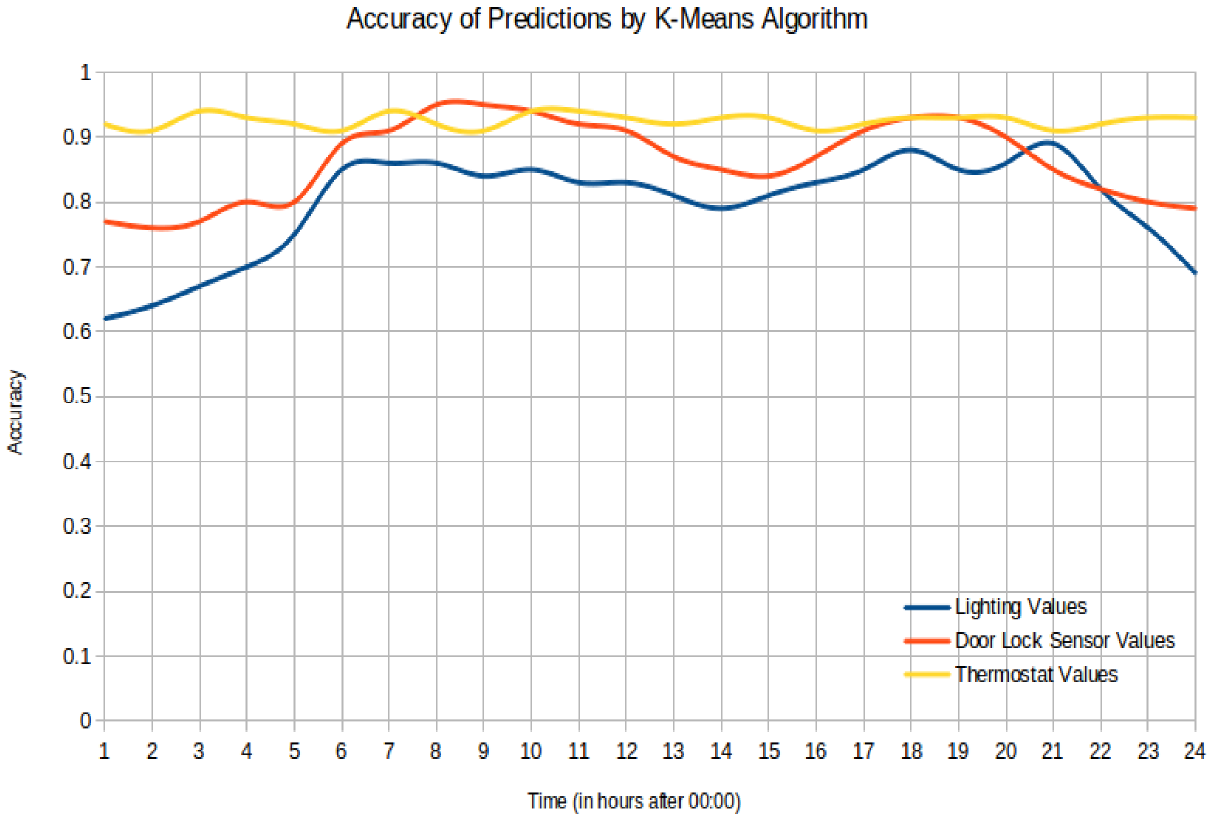

Illustration 3: Accuracy of Predictions by K-Means Algorithm

As with the K-NN Algorithm, It can be seen that for all three sensor value predictions, the time periods between 5:00 AM to 7:00 PM yielded the highest accuracy in recognizing good data from bad data. However, for all three sensor values, the K-Means Algorithm achieves much higher accuracy than the K-Nearest Neighbour Algorithm. Such an accuracy was achieved after running the test-suite multiple times with different starting means for the good and bad data. The primary reason for this was the iterative refinement technique used by the K-Means Algorithm. In such a technique, the centroid of each cluster becomes the mean after all values have been placed in clusters. Once the mean is changed, the values are reorganized into new clusters. This process is repeated until the centroid of the cluster is also the mean.

Overall, 40,560,300 different real and generated values from the three sensors were used in the test suite for the K-Means algorithm. The overall accuracy of the algorithm was 87.8%.

6. Results and Discussion

Both the objectives of performing this project were met. A smart home control system was created entirely in software, and closely simulated how an actual smart home control system implemented in hardware on Comcast Xfinity set-top boxes would work. Furthermore, the second objective of exploring different machine learning algorithm to drive the smart home control system was also fulfilled. Two different classes of machine learning algorithms, supervised and unsupervised learning, were explored via the K-Nearest Neighbour Algorithm and K-Means Algorithm. After extensive analysis of both algorithms, it has been proved that K-Means algorithm makes more accurate predictions of the nature of data as compared to the K-NN algorithm. This is mainly due to the iterative refinement process implemented in the K-Means algorithm.

The results of the project were slightly unexpected. Before the project was carried out, I had expected the K-NN algorithm to be more accurate, as it was a supervised learning algorithm. However, the unsupervised learning algorithm was more accurate due to iterative refinement. Furthermore, some of the various technologies used in the project were interesting. Twitter Storm proved to be a great resource in streaming hundreds of Gigabytes of data from spouts to bolts quickly and without any data loss, such the overhead to transfer data was a minimum. The bolts within the Storm topology were also able to execute both machine learning methods in a very efficient manner.

Overall, this project met all expectations and completed all objectives. The only part that would have been better to do differently is the part involving generating artificial or bad data. If the data that was generated was extremely similar to the real-world data when considering small sample sizes, then it would have been interesting to see how the accuracy of the K-NN and K-Means algorithms would have changed. Also, the opportunity to use other machine learning algorithms, such as the Fuzzy K-Means algorithm which allows data to be members of more than one cluster, would be present.

7. References

- Vainio, Antti M., Miika Valtonen, and Jukka Vanhala. *Learning and Adaptive Fuzzy Control System for Smart Home*. Tampere University of Technology, Institute of Electronics. Web. 7 May 2013.

- Guralnik, V., Haigh, K. Z., *Learning Models of Human Behaviour with Sequential Patterns*, Proceedings of the AAAI-02 workshop "Automation as Caregiver", pp. 24-30, 2002

- Kaila, L., Vainio, A.-M., Vanhala, J., *Connecting the smart home*, IASTED, Networks and Communication Systems, April 18-20, 2005, pp. 445-450
- Mäntyjärvi, J., Seppänen, T. *Adapting applications in handheld devices using fuzzy context information*, Interacting with Computers, vol. 15, issue 4, p. 521-538, August 2003
- Alonso, M., Malpica, J., Martinex de Agirre, A. Consequences of the Hughes Phenomenon on Some Classification Techniques. University of Alcalá, ASPRS 2011 Annual Conference, May 1-5, 2011.