USER PRIVACY INVASION: BY DISAGGREGATION OF AMI DATA

CS558 SPRING 2015

By Karthik Narisetti

TABLE OF CONTENTS

Contents

Introduction 1

Project Objective 2

Data set briefing and Approach 3

Usage and importance of Algorithams 4

Code snippets and Critical Ramblings 5

Results 8

Conclusion,Future work and References 10

PRIVACY INVASION: DISAGGREGATION OF AMI DATA

Introduction to AMI and related User Privacy issues

INTRODUCTION TO AMI INFRASTRCTURE

Advanced metering infrastructure (AMI) is an integrated system of smart meters, communications networks, and data management systems that enables two-way communication between utilities of installed vendors and customers who are using those smart meters to regulate the metering infrastructure. Customer systems include in-home displays, home area networks, energy management systems, and other customer-side-of-the-meter equipment that enable smart grid functions in homes, offices, and factories. Time-based rate programs include different types of electricity pricing options for customers that are made possible by AMI and sometimes include customer systems.

PRIVACY CONCERNS WITH SMART METER INFRASTRUCTURE

Although countless people claim that smart meters invade their privacy, it is unclear as to whether a compelling argument has yet been fully made that properly articulates an invasion of privacy assertion in a way that is understandable or convincing enough to persuade a sufficient number of legislators, governmental officials, court judges, or public utility commission hearing officers. To some extent, concerned consumers and activists have devoted a disproportionate amount of time and energy toward addressing potential health issues related to wireless smart meter RF emissions and have left the privacy arguments somewhat underdeveloped.

Many consumers claim that smart meters violate their constitutional right to privacy based upon the language in the Fourth Amendment to the U.S. Constitution where it states, **"The right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches, shall not be violated"**. In fact, many state constitutions are even more explicit in nature.

AMI – THE POSITIVE SIDE

The conventional electro-mechanical meter was a simple device with a single function – measure cumulative kWh usage to support the billing process. Meters to support Smart Grid will provide sensing and measurement capability to track much more information regarding both the usage and quality of power, and capability to support multiple rate forms. Smart Grid meters will also include communication capability that allow remote access by the utility and even some customers.

After the electric company has fully installed its advanced metering infrastructure, smart meters can benefit the electric customer by Offering **more detailed feedback on energy use**, Enabling them to **adjust their habits to lower electric bills** and Reducing blackouts and system-wide electric failures. It poses a wide range of uses including Providing **real-time data useful for balancing electric loads** and reducing power outages (blackouts), **Enabling dynamic pricing** (raising or lowering the cost of electricity based on demand), Avoiding the capital expense of building new power plants and Helping optimize income with existing resources.

Problem statement and Project Objective

POSSIBLE ACHIEVEMENTS BY ANALYSIS

This data not only helps utility companies make their businesses more efficient, but also helps consumers save money by using less energy at peak times. So, it is both economical and green. Smart meter infrastructure is fairly new to Utilities industry. As utility companies collect more and more data over the years, they may uncover further uses to these detailed smart meter activities. These are some of the uses of disaggregation of smart data.

- Invasion of privacy and intrusion of solitude
- Near real-time surveillance
- Behavior profiling
- Endangering the physical security of life, family, and property
- Unwanted publicity and embarrassment (e.g., public disclosure or private facts or the publication of facts which place a person in a false light)

PROBLEM STATEMENT

Clearly, Internet-based protocols, such as IPv4 and IPv6, which have been developed over many years, and which have widespread use, will provide a cost-effective baseline transport. Layering the suite of security protocols developed for IP [such as IPSec and Transport Layer Security (TLS)] on this baseline transport capitalizes on the vast work done in this area by protocol and industry experts. While the smart grid system is made up of a number of “energy” subsystems.

But **Data disaggregation is a major problem even though there are many communication layer level solutions** provided. Privacy concerns arise in this sense, if interested parties or even bad guys handle this kind of information. In fact, some researches show that it is possible to estimate personal information with a certain degree of accuracy, with relatively unsophisticated hardware and algorithms. In short, there is a need for discussion on the privacy aspect of data collection.

PROJECT OBJECTTIVE

The Objective of this project is to develop a data science methodology of load disaggregation of total house-hold (Group of house-holds) electric load into its end-uses by employing by using **Steady State signature**, **Frequency analysis** and **Clustering**.

Data set and Approach

MACHINE LEARNING DATA SET USED

I am using a data set that is measured at UCI (Center for machine learning and IS).

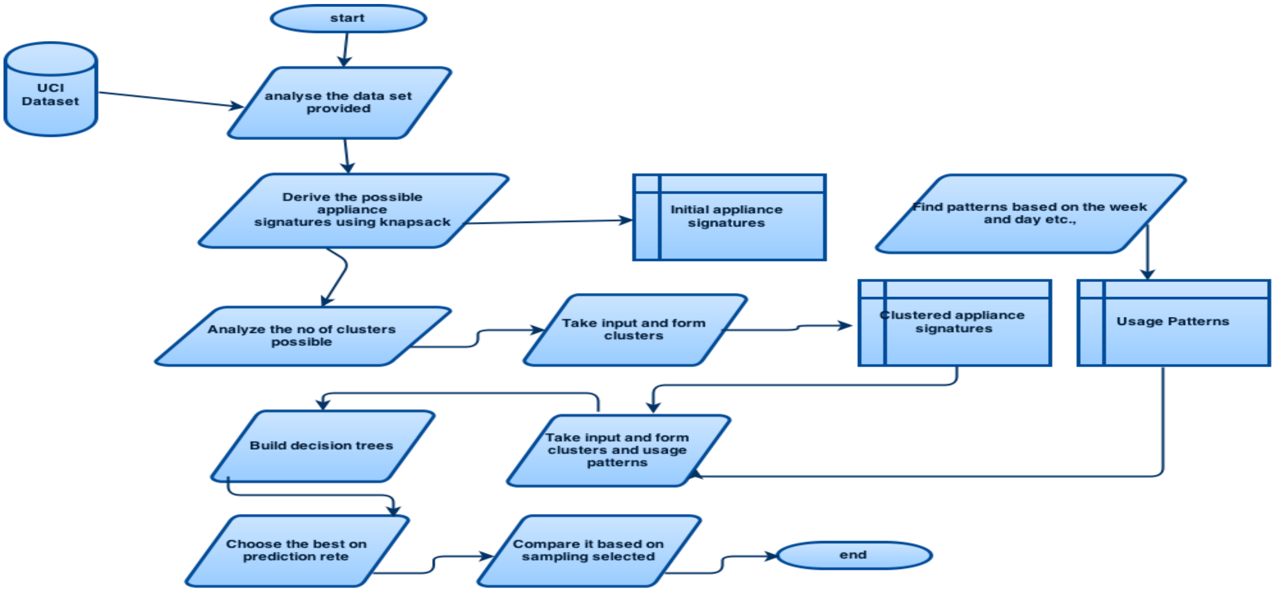
- 20,75,259 Measurements of electric power consumption in one household
- A one-minute sampling rate
- A period of almost 4 years
- Different electrical quantities
- Some sub-metering values are available (3 meters in data set)

The appliances that the data set were built on are as below.

MY APPROACH ON DATA DISAGGREGATION

In this project in order to achieve the goal we employ Knapsack algorithm to retrieve the individual appliance's power consumption signatures to indicate which appliance is on or off at a given instance of time and K means clustering to cluster the result of knapsack algorithm into group of appliances which are usually on at the same time interval.

FLOW CHART FOR MY APPROACH


start
UCI Dataset
analyse the data set provided
Derive the possible appliance signatures using knapsack
Initial appliance signatures
Find patterns based on the week and day etc.,
Analyze the no of clusters possible
Take input and form clusters
Clustered appliance signatures
Usage Patterns
Build decision trees
Take input and form clusters and usage patterns
Choose the best on prediction rete
Compare it based on sampling selected
end

Usage and Importance of Algorithms

ALGORITHMS EXPLAINED

When you have a document that shows a lot of numbers, it’s a good idea to have a little text that explains the numbers. You can do that here.

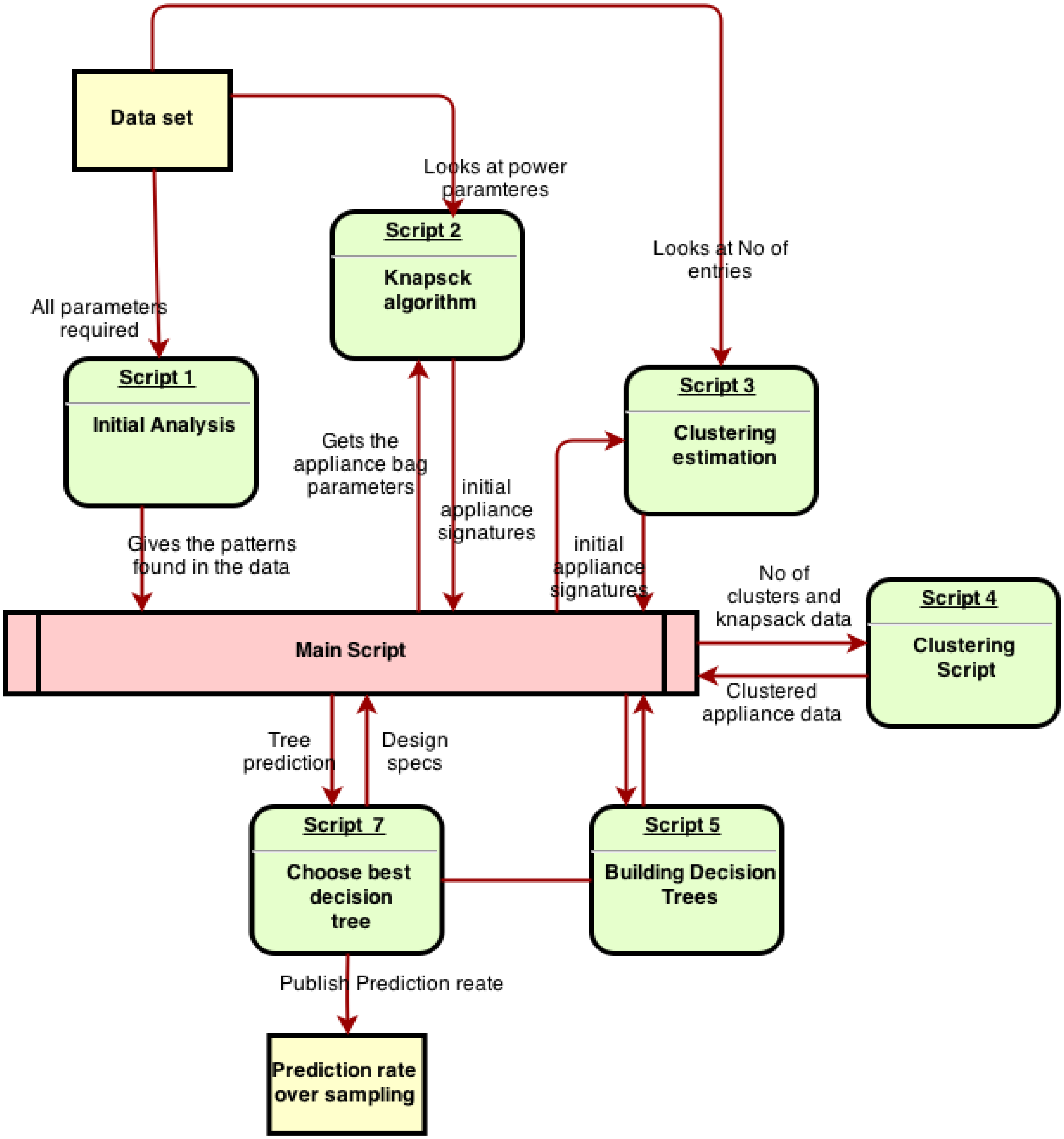
0/1 Knapsack is a combinatorial optimization algorithm which given a set of items, each with a mass and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible. It derives its name from the problem faced by someone who is constrained by a fixed-size knapsack and must fill it with the most valuable items. We **use this algorithm to help us retrieve the appliances that are ‘ON’ at a given point of time from the input aggregate data of power consumption of a regular house hold**. Using these power consumption signatures, the algorithm, tries to fit the maximum number of appliances within the Actual power consumption in KW from the aggregated input of the household for that given time instance. Employing this method we can derive which appliances are actually ‘ON’ at a given instance of time to use it for further processing.

K-means clustering is a method of vector quantization, popular for cluster analysis in data mining. K-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. We use **this algorithm to help us cluster the output of the knapsack algorithm**, which is the appliances that are ON at a given instance of time. K means clusters the appliances to groups of clusters which show all the other appliances which are also ON along with it. This can be used for both good, an Electric service company attempting to save up energy during the peak hours and also save money for the consumers as well as nefarious purposes as a thief using this to evaluate the patterns and using them to his advantage, to maybe attack when the house owners are sleeping or are away.

Random forests is an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the class's output by individual trees. These are combination of **tree predictors such that each tree depends on the values of a random vector sampled independently** and with the same distribution for all trees in the forest. The generalization error for forests converges as to a limit as the number of trees in the forest becomes large. **CART (classification and regression tree)** was used in this greedy, top-down binary, recursive partitioning, that divides feature space into sets of disjoint rectangular regions.

CODE/ DATA FLOW

For coding I followed ESB architecture where the main script acts as a bus to let the data flow between various scripts.



KNAPSACK FUNCTION

```r
knapsack <- function(value, weight, limit){
  benefit.to.cost <- value / weight #Create ratio
  df = data.frame(value, weight, benefit.to.cost) # turn it into a DF
  df <- df[with(df, order(-benefit.to.cost)), ] # Sort by benefit.to.cost
  rownames(df) <- NULL # Reset the row names for easier indexing
  df$total.weight <- ifelse(cumsum(df$weight) <= limit, cumsum(df$weight), 0) # Add first items that fit
  # I need to add a break here if nothing fits in the bag on the first pass
  for(i in 2:nrow(df)){ #Start in row 2 because some values have been added above
    df$total.weight[i] <- ifelse(df$weight[i] + df$total.weight[i-1] <= limit, # If adding won't go over limit
                                 df$weight[i] + df$total.weight[i-1], df$total.weight[i-1]) # If it will, keep Weight the same
  }
  df$add <- 0
  df$add[1] <- ifelse(df$total.weight[1] > 0, 1, 0)
  for(i in 2:nrow(df)){ #Start in row 2
    df$add[i] <- ifelse(df$total.weight[i] > df$total.weight[i-1], 1, 0) # 1 if it has been added
  }
  return(df)
}
```

CHOOSING A BEST DECISION TREE

```r
#it extracts the best tree number
extractBestTree <- function(model, prediction, startsAt = 0, y = NULL){

  if(model$distribution == 'bernoulli' | model$distribution == 'adaboost' | model$distribution == 'huberized'){
    predictionVector <- prediction[ , which.min(abs(n.trees - which.min(model$train.error)))]
    predictionVector <- round(predictionVector)
    predictionVector[predictionVector < 0] <- 0
    predictionVector[predictionVector > 1] <- 1

  }else if(model$distribution == 'multinomial' & length(dim(prediction)) == 2 & !is.null(y)){
    probablities <- table(y) / length(y)
    predictionOne <- apply(prediction, 1, function(vector){
      return(vector *  probablities)
    })
    if(startsAt == 0){
      predictionVector <- apply(predictionOne, 2, which.max) - 1 # the minus one is because the first index refers to zero
    }else{
      predictionVector <- apply(predictionOne, 2, which.max) # the minus one is because the first index refers to zero
    }

  }else if(model$distribution == 'multinomial' & length(dim(prediction)) == 2 & is.null(y)){
    if(startsAt == 0){
      predictionVector <- apply(prediction, 1, which.max) - 1 # the minus one is because the first index refers to zero
    }else{
      predictionVector <- apply(prediction, 1, which.max) # the minus one is because the first index refers to zero
    }

  }else{
    predictionOne <- prediction[ , , which.min(abs(n.trees - which.min(model$train.error)))]

    if(startsAt == 0){
      predictionVector <- apply(predictionOne, 1, which.max) - 1 # the minus one is because the first index refers to zero
    }else{
      predictionVector <- apply(predictionOne, 1, which.max) # the minus one is because the first index refers to zero
    }
  }
  return(predictionVector)
}
```

SOME RAMBLINGS IN CODING AND PREDICTION ANALYSIS

The challenge is to take this structured data, synthesize it, quantify it and to increase its selling value by moving from historical analysis to predictive analysis. This use case is a best demonstration for how Apache Hadoop can help fueling analytics of enormous data.

- It took 9 hours, at most of time more than CPU consumption of 78% most of time.
- After using single node cluster R on Hadoop it took more than 6 1/2 hours of time to run (on a 4-core dummy node cluster).
- The largest vector is of 358GB in happened in clustering the appliances.
- 2 Node cluster gave me an advantage of 5 hours where expected is less than 4 hours.
- To get the data from HDFS and write it back to HDFS would need some changes in design and I did most of them added some re-work.
- There are a lot of random patterns in this kind of data set.
- It is very difficult to derive to a conclusion from a machine learning data set unless you have a strong ground truth.
- Generalizing is not as accurate when registered appliance signatures were used.

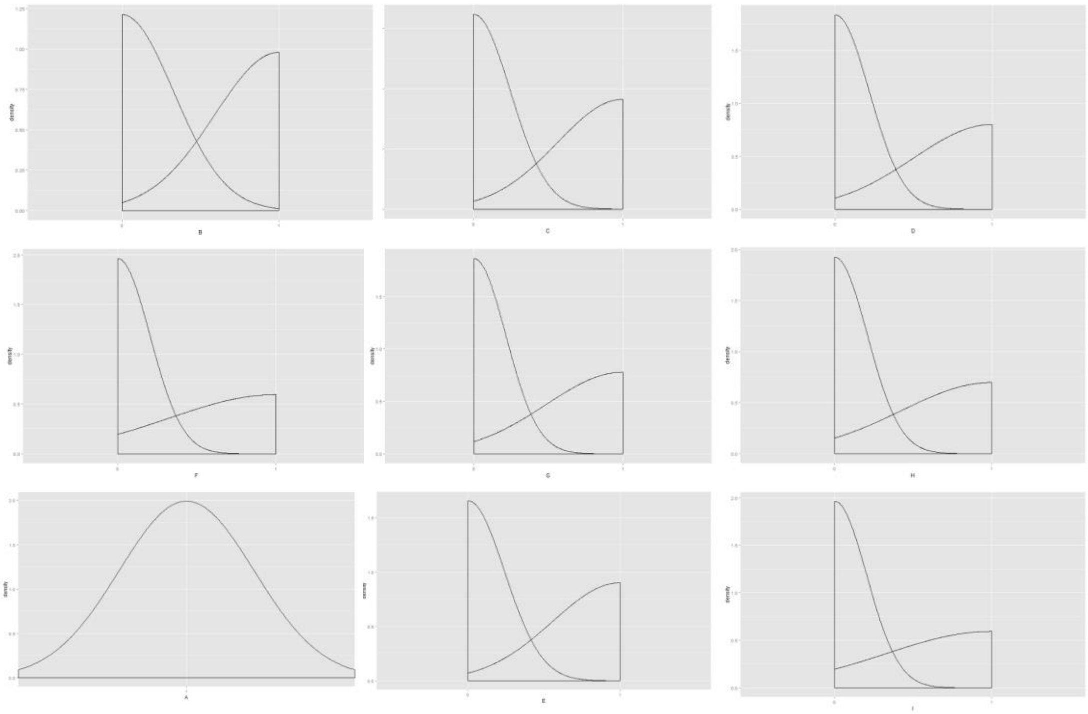
Results

PLOTS

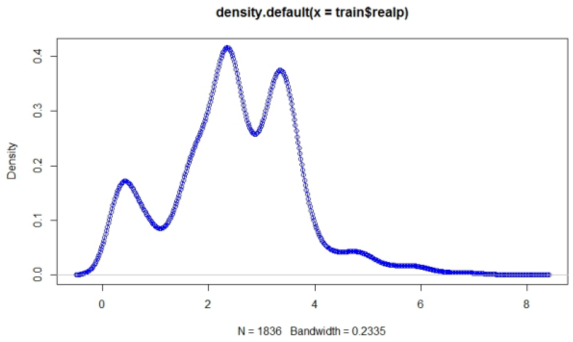
As most of the things involved in my project are plots, I will be providing them here.

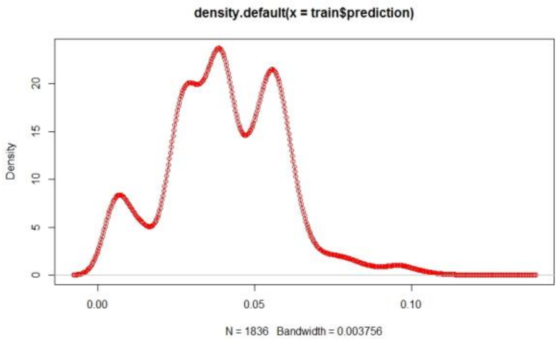
KERNEL DENSITY GRAPHS OF EACH APPLIANCE

The Kernel density estimation (KDE) is a non-parametric way to estimate the probability density function of a random variable (Here it is state of an appliance). Kernel density estimation is a fundamental data smoothing problem where inferences about the population are made, based on a finite data sample. (Helps in predicting which can go together).

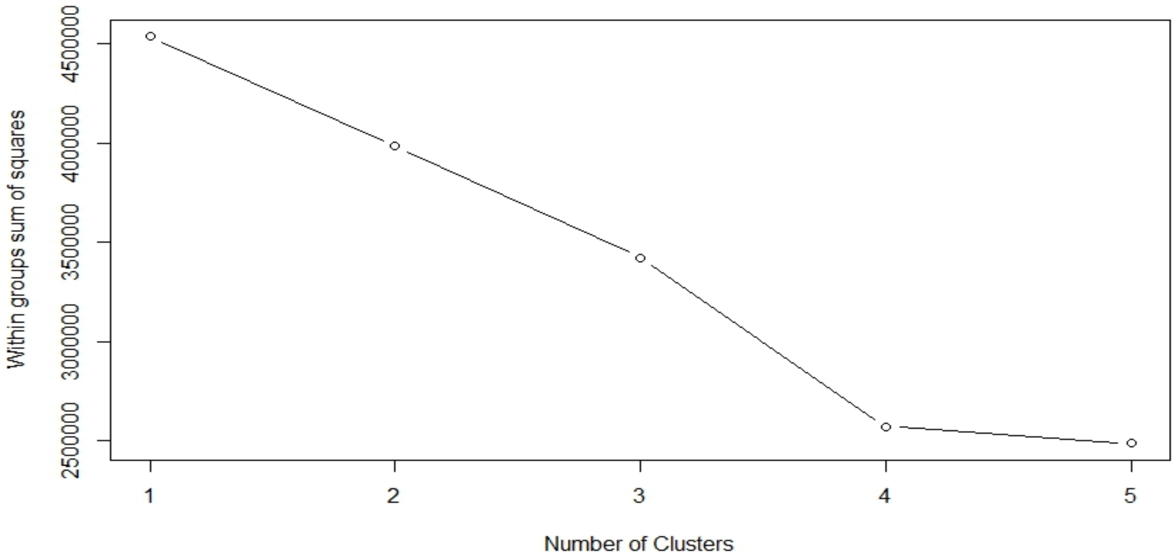


PREDICTION PLOTS (ACTUAL VS PREDICTED)

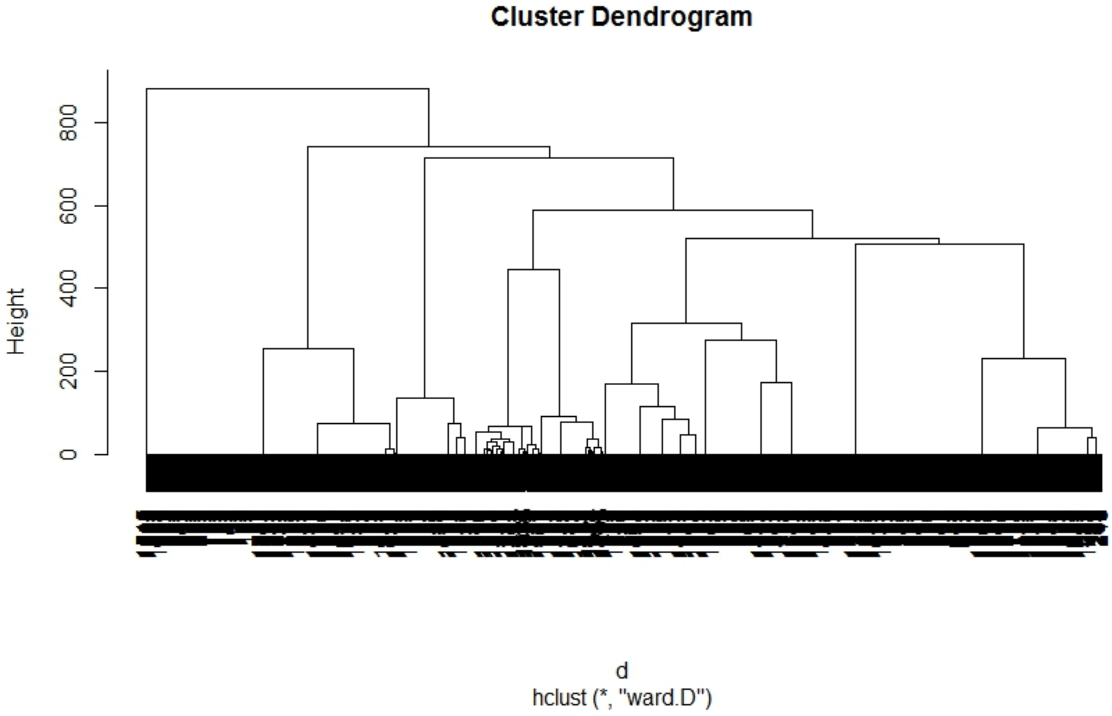




CLUSTER ESTIMATION OVER THE PROVIDED SAMPLES



DENDOGRAM OF PROVIDED CLUSTERS (5 IN NUMBER)



Conclusion

- Am able to get around **58% of predictions correct.**
- With different decision trees I improved the prediction at a maximum by **0.85-2.5%.**
- So, at any point of time my algorithm is able to predict the right pattern by **+59 %**

FUTURE WORK

- A defense mechanism is under implementation.
- Identified patterns need to be suppressed in order to mask the appliance signature.
- I will be adding a noise (Based on the determination of most active/passive appliance at a point of time)
- Provide results after implementation.

REFERENCES

- Algorithm AS 136: A K-Means Clustering Algorithm J. A. Hartigan and M. A. Wong Journal of the Royal Statistical Society. Series C (Applied Statistics) Vol. 28, No. 1 (1979), pp. 100-108
- RANDOM FORESTS Leo Breiman Statistics Department University of California Berkeley, CA 94720
- Dietterich, T. [1998] An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting and Randomization, Machine Learning 1-22
- Kaggle - https://www.kaggle.com/
- Combining Factorization Model and Additive Forest for Collaborative Followee Recommendation Tianqi Chen, Linpeng Tang, Qin Liu, Diyi Yang, Saining Xie, Xuezhi Cao, Chunyang Wu, Enpeng Yao, Zhengyang Liu, Zhansheng Jiang, Cheng Chen, Weihao Kong, Yong Yu ACMClass@SJTU Team, Shanghai Jiao Tong

ONE LAST THING!
THANKS TO YOU!!☺
PROFESSOR KEVIN