Automated Prompting in a Smart Home Environment

Authors:

Barnan Das, Chao Chen, Nairanjana Dasgupta & Diane J. Cook

Center for Advanced Studies in Adaptive Systems (CASAS)
Washington State University
Pullman WA, USA

WASHINGTON STATE UNIVERSITY

WSU CASAS

Outline

- Introduction
- Data Collection
- Dataset and Performance Measures
- Statistical Analysis
- Learning Models
- Sampling
- Improvement of Learning Algorithms
- Future Work

Outline

- **Introduction**
- Data Collection
- Dataset and Performance Measures
- Statistical Analysis
- Learning Models
- Sampling
- Improvement of Learning Algorithms
- Future Work

The PUCK (Prompting Users and Control Kiosk)

Goal: To identify when an activity step has been missed or performed erroneously and deliver an appropriate prompt when one is required.

Related work:

I) Rule based (time and context)
II) Reinforcement Learning
III) Planning
IV) Supervised Learning

How different is The PUCK?
Unlike most of the early and modern reminder systems, it doesn't rely on direct or indirect user feedback.

Reason: In a real world smart environment dealing with cognitively impaired people, it would not be possible to rely or even receive user feedback.

Outline

- Introduction
- **Data Collection**
- Dataset and Performance Measures
- Statistical Analysis
- Learning Models
- Sampling
- Improvement of Learning Algorithms
- Future Work

Testbed

Two story apartment – Living room, dining area, kitchen on 1st floor and 3 bedrooms and bathroom on the 2nd.

Sensors:
- Motion sensors on the ceiling
- Door sensors on the apartment entry and doors of cabinet, refrigerator and microwave oven
- Temperature sensor in each room
- Power meter
- Analog sensors for burner and water usage
- Sensor to keep track of telephone use.

One bedroom used as a control room to monitor activities.
Monitoring: Webcam
Prompting: Audio delivery system.

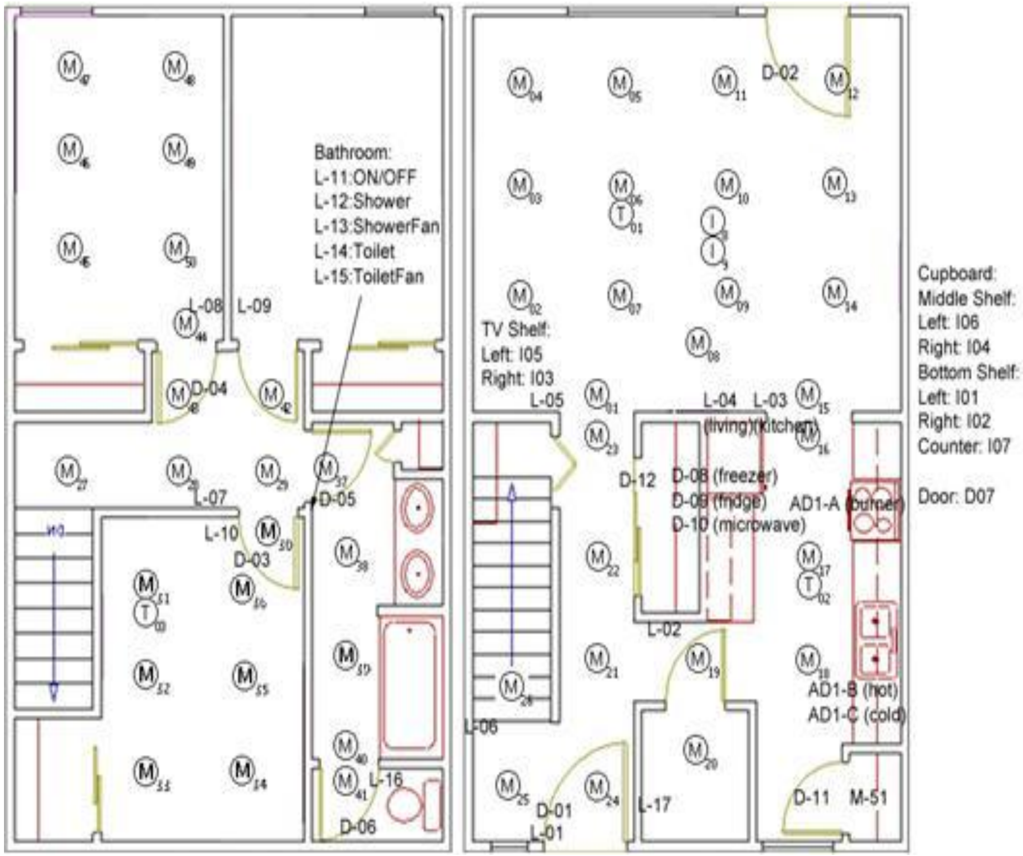


Figure 1. Three-bedroom smart apartment used for data collection (Sensors: motion (M), temperature (T), water (W), burner (B), telephone (P) and item (I)).

Experimentation Methodology

Experiments done on 8 ADLs:

1. Sweeping	2. Medication	3. Birthday Card	4. DVD
5. Water Plants	6. Phone Call	7. Cooking	8. Select Outfit

Activities are subdivided in the following way:

Cooking

1. Participant retrieves materials from cupboard.
2. Participant fills measuring cup with water.
3. Participant boils water in microwave.
4. Participant pours water into cup of noodles.
5. Participant retrieves pitcher of water from refrigerator.
6. Participant pours glass of water.
7. Participant returns pitcher of water.
8. Participant waits for water to simmer in cup of water.
9. Participant brings all items to dining rooms table.

Experimentation Methodology

Procedure for conducting experiment:

Step 1: Participants are asked to perform activities in the smart apartment.

Step 2: When an activity is about to begin, an instruction (with basic details of the activity) is delivered to the participant.

Step 3: The participant, while going through the steps of the activity, is given a prompt in the following conditions:

- The participant performs steps for other activities and not the current one.
- Steps irrelevant to the current activities are being performed.
- A specified amount of time has elapsed since the initiation of the current step.

Step 4: After the activity is completed, the participant is asked a set of questions that help in determining the difficulties they might have faced while doing the activity or following the prompts.

Step 5: The instruction of the next activity is given and the same process continues.

Annotation

Sample of sensor events:

Date	Time	Sensor ID	Message
2009-02-06	17:17:36	M45	ON
2009-02-06	17:17:40	M45	OFF
2009-02-06	11:13:26	T004	21.5
2009-02-07	11:13:26	P001	747W
2009-02-09	21:15:28	P001	1.929kWh

Annotated sensor events. Step is represented by: <Activity ID>.<Step Number>

2009-05-11	14:59:54.934979	D010	CLOSE	7.3
2009-05-11	14:59:55.213769	M017	ON	7.4
2009-05-11	15:00:02.062455	M017	OFF	
2009-05-11	15:00:17.348279	M017	ON	7.8
2009-05-11	15:00:34.006763	M018	ON	7.8
2009-05-11	15:00:35.487639	M051	ON	7.8
2009-05-11	15:00:43.028589	M016	ON	7.8
2009-05-11	15:00:43.091891	M015	ON	7.9
2009-05-11	15:00:45.008148	M014	ON	7.9

Feature Generation

Feature #	Feature Name	Description
1	stepLength	Length of the step in time (seconds)
2	numSensors	Number of unique sensors involved with the step
3	numEvents	Number of sensor events associated with the step
4	prevStep	Previous step
5	nextStep	Next step
6	timeActBegin	Time (seconds) elapsed since the beginning of the activity
7	timePrevAct	Time (seconds) difference between the last event of the previous step and the first event of the current step
8	stepsActBegin	Number of steps visited since the beginning of the activity
9	activityID	Activity ID
10	stepID	Step ID
11	M01 … M51	All of M01 to M51 are individual features denoting the frequency of firing of these sensors associated with the step
12	Class	Binary class. 1-”Prompt”, 0-”No-Prompt”

Outline

- Introduction
- Data Collection
- **Dataset and Performance Measures**
- Statistical Analysis
- Learning Models
- Sampling
- Improvement of Learning Algorithms
- Future Work

- The learning model is trained with the data collected from 20 participants.
- Total number of steps: 53
- Recognizable by annotators: 38
- The participants were delivered prompts in 39 cases involving any of the 38 recognizable steps.
- Approximately 8% of the total instances are positive and the rest are negative.

Therefore, accuracy cannot be a good performance measure.

True Positive (TP) Rate: Percentage of activity steps that are correctly classified as requiring a prompt.

True Negative (TN) Rate: Percentage of activity steps that are correctly classified as not requiring a prompt.

Area Under ROC Curve: It considers the performance of the classifiers without taking into account class distribution or error cost.

Outline

- Introduction
- Data Collection
- Dataset and Performance Measures
- **Statistical Analysis**
- Learning Models
- Sampling
- Improvement of Learning Algorithms
- Future Work

Selecting Feature Subset for Analysis

Only statistically relevant features are selected for analysis:

1. stepLength
2. numSensors
3. numEvents
4. timeActBegin
5. timePrevAct

Simple Statistical Information

Feature	Mean	Std Dev	Sum	MIn	Max
stepLength	71.57	137.31	36930	1	815
numSensors	1.98	1.46	1021	1	9
numEvents	4.80	5.70	2479	1	45
timeActBegin	82.29	122.63	42464	0	864
timePrevAct	12.92	32.64	5003	0	323

Pearson Correlation Coefficients

	stepLength	numSensors	numEvents	timeActBegin	timePrevAct
stepLength	1	0.21	0.74	-0.047	-0.083
numSensors	0.21	1	0.59	0.1	-0.05
numEvents	0.74	0.59	1	0.043	-0.07
timeActBegin	-0.047	0.1	0.043	1	0.27
timePrevAct	-0.083	-0.05	-0.08	0.27	1

minimum-Redundancy-Maximal-Relevance (mRMR) Feature Selection

Uses mutual information to find the relation betweens features which have maximum relevance and minimum redundancy.

We get the following features in order of there ranking:

1. stepLength
2. numSensors
3. timeActBegin
4. timePrevAct

Box Plot

Purpose of Statistical Analysis: Finding outliers.

The data is ***not*** normally distributed. So mean and standard deviation are not good measures to find outliers.

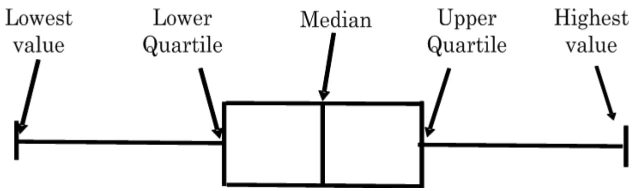


Figure 2. Configuration of a Box Plot

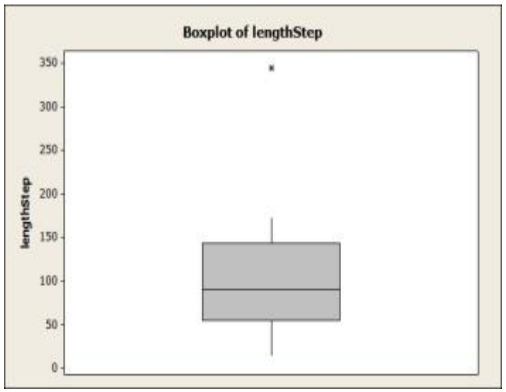
Reason for using Box Plot:
Displays the difference between populations without making any assumption of the underlying statistical distribution and the distance between the different parts of the box help indicate the degree of spread and skewness in the data set.

In this paper, box plot is used to find the outliers which are participants who performed a step of an activity in an ***unusual*** way.

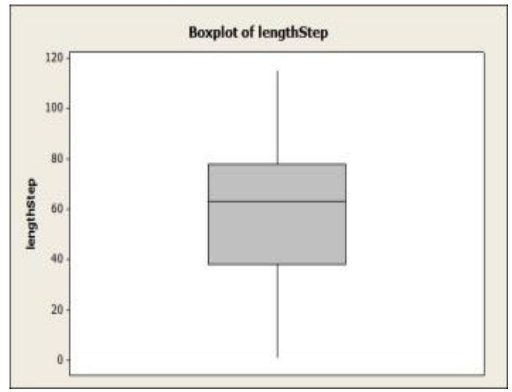
Any data point farther than 1.5*IQ (where IQ = Upper Quartile – Lower Quartile) is an outlier.

Experimental Results

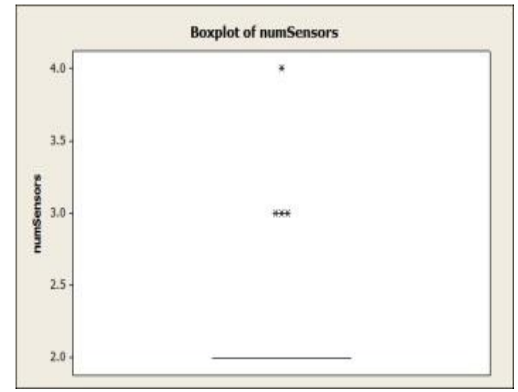
The outliers for all the features associated with every step is detected by the Box Plot.



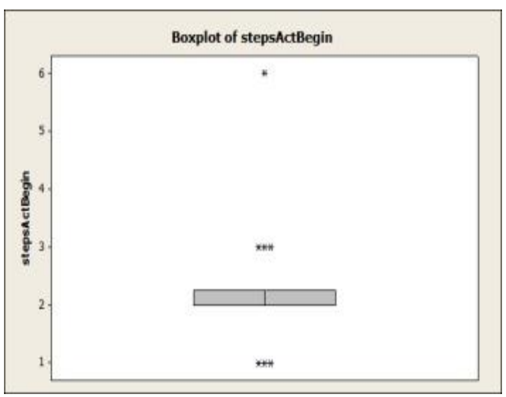
a) Step:1.6, Feature:stepLength



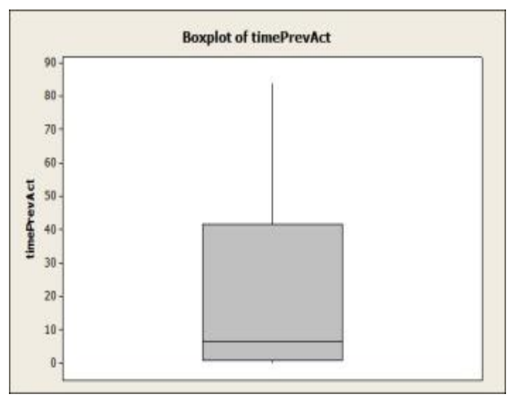
b) Step:8.2,Feature:stepLength



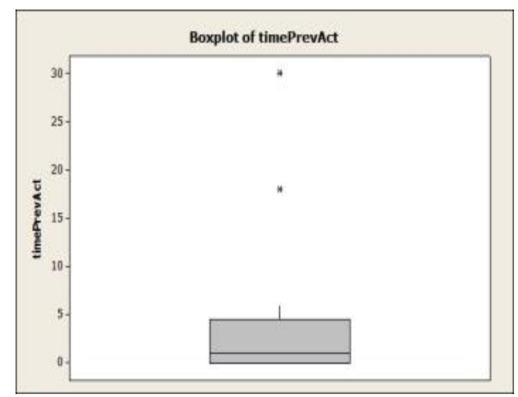
c) Step:8.3,Feature:numSensors



d) Step:1.4, Feature:stepActBegin



e) Step:2.2,Feature:timePrevAct



f) Step;7.7,Feature:timePrevAct

Figure 3. Box Plot for some steps and features

Experimental Results

All the features are capable of detecting the outliers, but there is no single "silver bullet" feature that defines all error based outliers in activity steps.

We consider a combination of these features to deduce a score based on ranked weight obtained from mRMR.

$$p = w_1 f_1 + w_2 f_2 + \dots + w_n f_n$$

where n is the total number of features ($n=4$ in our case), f_i for $1 \le i \le n$, is 1 if the value f_i is an outlier for the corresponding box plot and 0 otherwise; w_i for $1 \le i \le n$, are the weights assigned to each feature as per the rank. For some x, such that $0 \le x \le 1$, an instance is marked as an outlier if $p \ge x$.

Evaluation of the performance of this outlier detection as a classifier is done by comparing the prediction of every instance with the original "prompt" steps.

Experimental Results

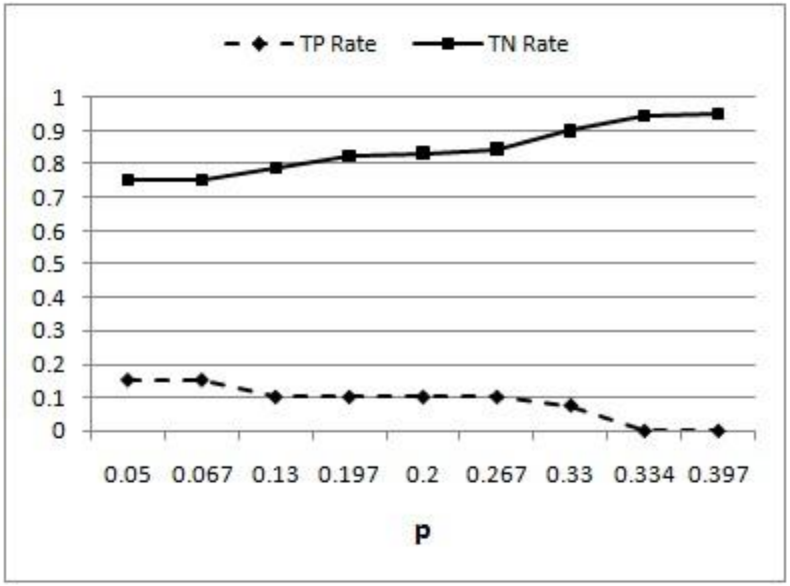


Figure 4. Performance of outlier detection

Outline

- Introduction
- Data Collection
- Dataset and Performance Measures
- Statistical Analysis
- **Learning Models**
- Sampling
- Improvement of Learning Algorithms
- Future Work

Prediction Models Used

- Decision Tree
- k-Nearest Neighbor
- Boosting

Experimentation with Original Dataset

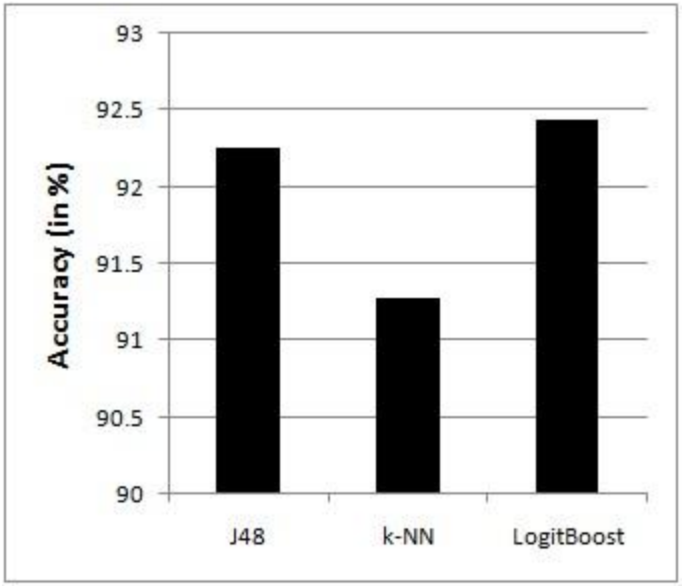


Figure 5. Accuracy (in %) for J48. K-NN and LogitBoost

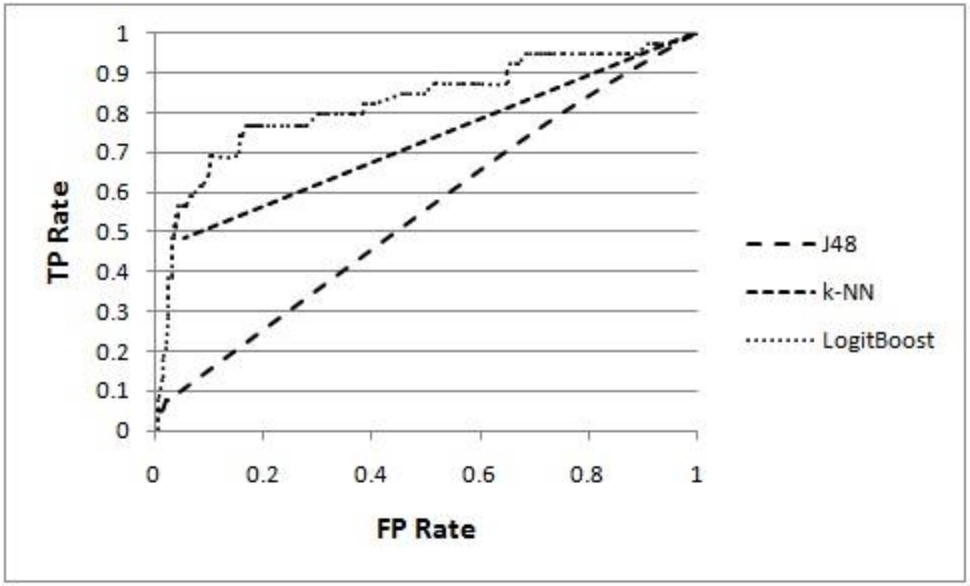


Figure 6. ROC Curves

	TN Rate	TP Rate
J48	0.994	0.051
kNN	0.948	0.487
LogitBoost	0.977	0.282

Failure of Learning Algorithms on Original Dataset

- The data is highly skewed towards negative class.
- Reason: The purpose of The PUCK is not to prompt an inhabitant in every step of an activity but to deliver the prompt only for steps where individuals need the help to complete the task.
- Decision trees do not take all attributes into consideration to form a hypothesis. The inductive bias is to *prefer a shorter tree over larger trees*.
- kNN estimates the target function locally and differently for each new instance to be classified. Therefore, the performance is shade better than others.
- A boosting technique works well for unstable learning algorithms (whose output classifier undergoes major changes in response to small changes in the training data.)

Outline

- Introduction
- Data Collection
- Dataset and Performance Measures
- Statistical Analysis
- Learning Models
- **Sampling**
- Improvement of Learning Algorithms
- Future Work

Re-balancing the dataset synthetically.

Done using two methods:

1. Undersampling: The majority class is shortened by ignoring some of its instances.

2. Ovesampling: The number of instances in the minority class is increased by some means.

Drawbacks:

Undersampling: Can throw away potentially useful data.

Oversampling: Can cause the classifier to overfit as a common technique is to replicate data and thus letting the classifier to formulate rules on insufficient or replicated data.

SMOTE
(Synthetic Minority Over-Sampling Technique)

A combination of both under-sampling and over-sampling, where over-sampling is not done by just replicating the positive instances but by the generation of new instances which are similar to others.

Over-sampling is done by taking each minority class sample and synthesizing a new sample by randomly choosing any/all of its k minority class nearest neighbors.

Generation of the synthetic sample is done in the following way:

i. Compute the difference between the feature vector (sample) under consideration and its nearest neighbor.
ii. Multiply this difference by a random number between 0 and 1.
iii. Add the product to the feature vector under consideration.

SMOTE-Variant

Over-sampling:

1. Randomly pick an instance from the minority class.
2. Nearest neighbor is calculated by considering minority class instances which have same activityID and stepID.
3. One of the neighbors is randomly chosen and a new instance is synthesized in the same way SMOTE does.

Under-sampling:

Randomly choosing a sample of size k from the entire population without repetition.

SMOTE-Variant

Algorithm: SMOTE-Variant (T, N, M)

Input: All the instances of minority class stored in list T, Number of minority class samples N, Desired number of minority class samples M
Output: M minority class samples stored in list S

```
if (M > N)
    then S = T        //S stores all the instances in T
    for i ← 1 to M – N
        t = randomize(T) //Randomly chosen instance from T
        neighbor[] = null //List storing all the nearest  neighbors of instance t
        for j ← 1 to length(T)
            if (t.activityID==T[j].activityID) AND (t.stepID==T[j].stepID)
                neighbor[].append(T[j]) // Appending the list with the neighbor
        endfor
            s = randomize(neighbor) //Randomly choosing an instance from list neighbor
            diff = t – s //Both t and s are vectors, so this is a vector subtraction
            rand = random number between 0 and 1
            newInstance = t  + diff  *  rand
            S.append(newInstance)
    endfor
endif
```

Effect of Class Distribution

Natural class distribution is not the best for training a classifier.

Given plenty of data, the optimal distribution generally contains 50% to 90% of the minority class instances.

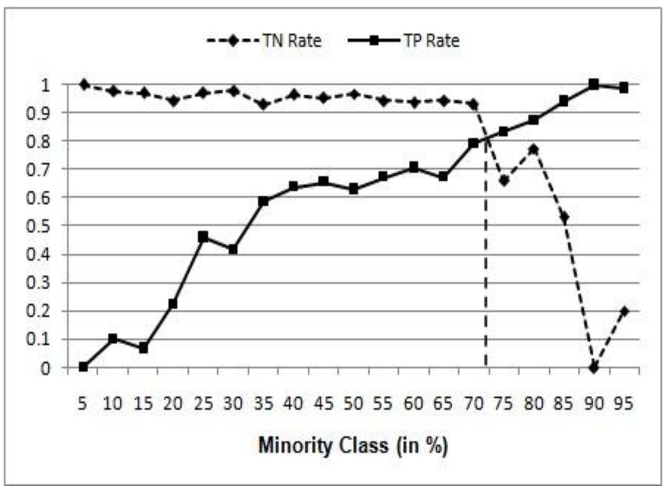


Figure 7. TN and TP Rates for different class distributions

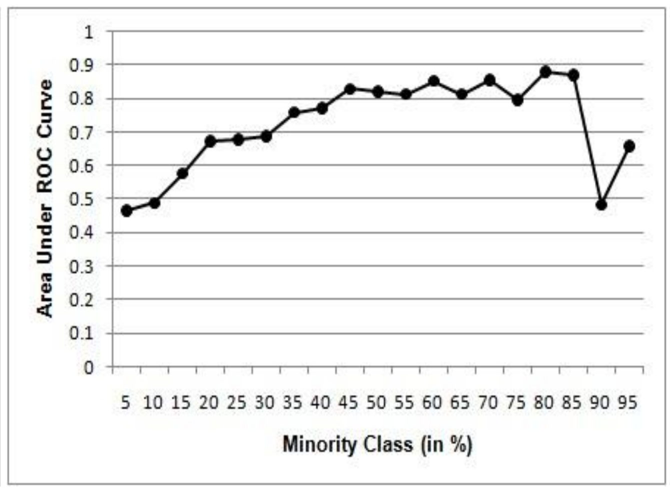


Figure 8. AUC for different class distributions

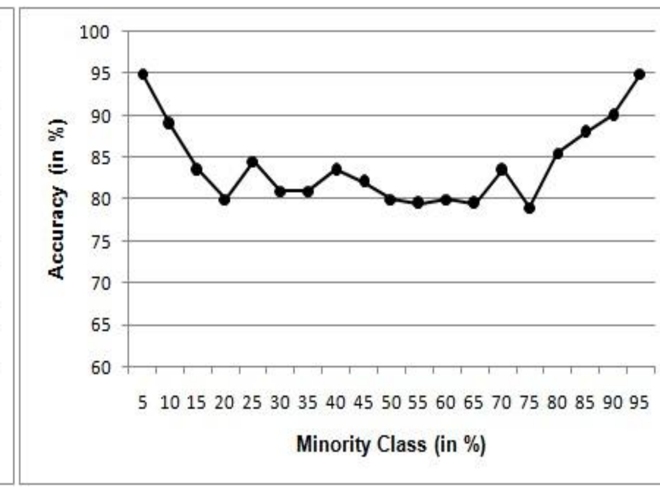


Figure 9. Accuracies for different class distributions

Outline

- Introduction
- Data Collection
- Dataset and Performance Measures
- Statistical Analysis
- Learning Models
- Sampling
- **Improvement of Learning Algorithms**
- Future Work

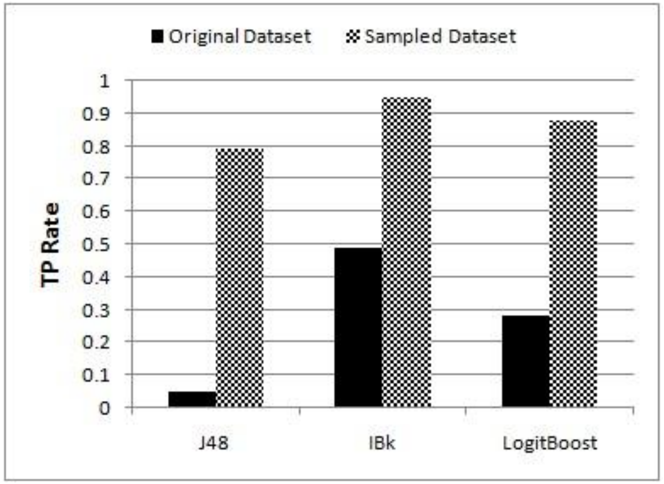


Figure 10. Comparison of TP Rates

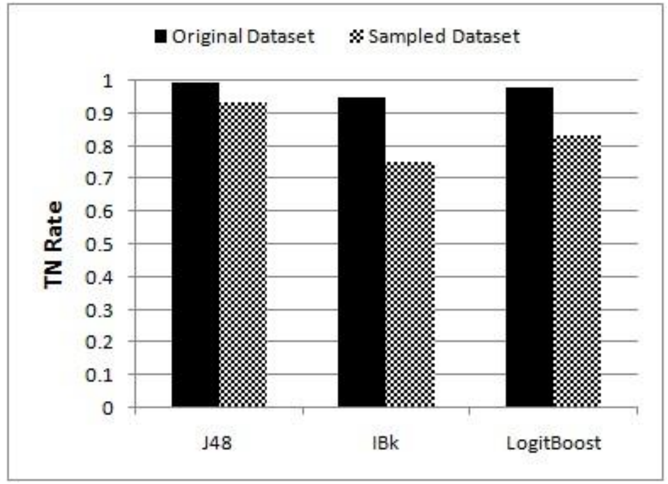


Figure 11. Comparison of TN Rates

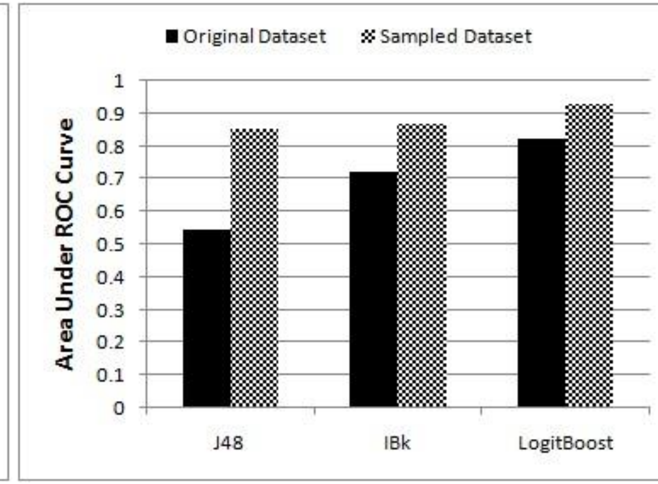


Figure 12. Comparison of AUCs

	Original Dataset	Sampled Dataset
J48	92.25	83.5
k-NN	91.28	89.0
LogitBoost	92.44	86.5

- Dealing with scanty data by using Self Training and comparative analysis with current work.
- Real time prompting, where predictions would be done on event level rather than step level (as done in current work).

Questions?