# Effect of Meditation on Brain Training

Myo W. Thane, Jaskiran Sur, Deisharrah Allen-Benson, Anh Do, Advisor: Professor Hong Lin[1]
[1]Department of Computer Science and Engineering Technology
[2]Center for Community Engagement and Service Learning
University of Houston-Downtown

## Abstract

Scientists and researchers have been studying the benefits of meditation on the human brain and body. An enthused group of UHD students, led by an inspired professor, studies the mindfulness intervention of meditation on brain training using scientific machine-learning methods and EEG (Electroencephalogram) analysis.

The project collects EEG signals from volunteers in 3 stages. At the pre-meditation stage, volunteers play online brain training games while EEG signals of their brain and game scores are collected. In the meditation stage, the volunteers play a virtual reality meditation game for 10 minutes while EEG signals are collected. At the post-meditation stage, the volunteers play the same brain training games that they previously played while EEG signals and game scores are collected. The scores recorded during the pre-meditation and post-meditation stages are compared, while collected EEG data are processed using multiple machine-learning methods. On average, the post-meditation score of the volunteer is higher than the pre-meditation score by around 15%. Various machine learning methods also return high accuracies in classifying the EEG data read in 3 stages.

This study gives positive indicators that meditation can enhance brain training performance and sheds light on the further development of contemplative practices for enhancing learning effectiveness.

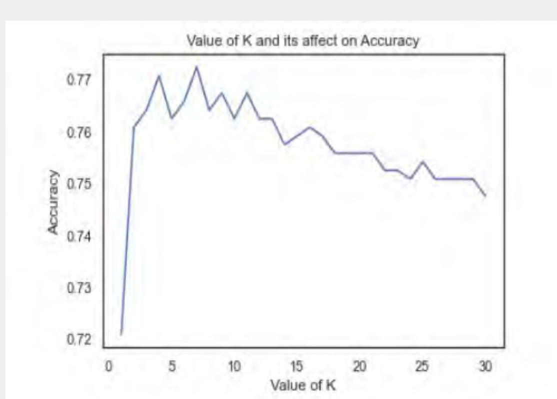
## Data Analysis Procedures

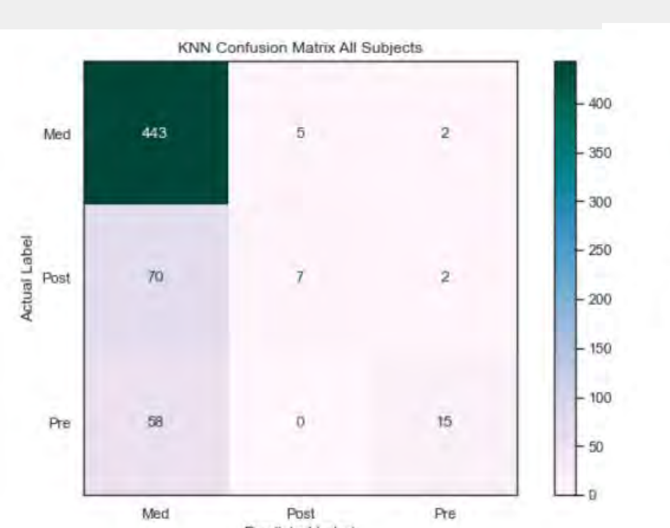
- Load the data in each of the Matlab files into a data frame, resulting in 3 data frames. Each data frame has 34 channels of EEG samples.
- Clean each dataset by removing the zero fill-in segments.
- Compute the alpha PSIs of 34 channels, using 1024 as the segment size, as depicted above, and save the alpha PSIs in a data frame of 34 columns. Repeat this for each of the 3 data frames.
- Create a list of brain activity labels, corresponding to each data frame, as depicted above.
- Combine 3 data frames into one by vertical stacking.
- Create a correlation coefficient matrix of the alpha PSIs of the 34 channels.
- Remove co-linearity from the combined data frame, as depicted above.
- Print the correlation coefficient matrix of the alpha PSIs of the remaining channels.
- Normalize data in every remaining channel using the min-max normalization.
- Randomly sample 80% of each dataset and save them in a training dataset, and save the remaining 20% of each dataset in a testing dataset. Split the list of labels accordingly.
- Train the machine learning model using the training dataset.
- Use the model to predict the target feature of the testing dataset.
- Create a confusion matrix to compare the predicted state activities to the actual activities and compute the accuracy.

## KNN and Naive Bayes

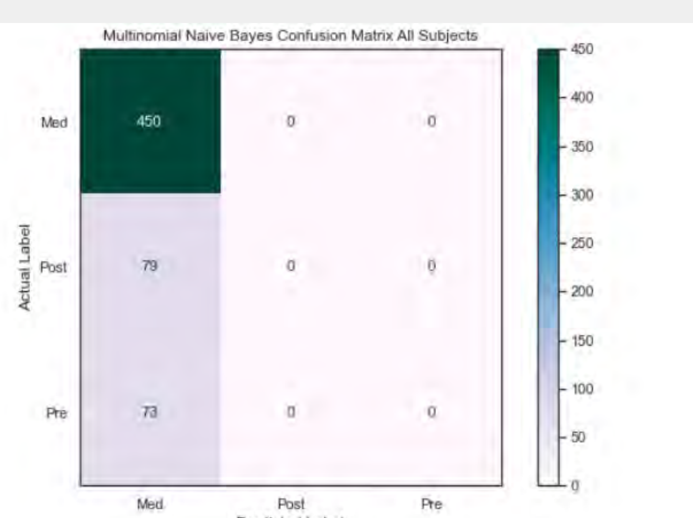
**KNN or K-Nearest Neighbors** is a supervised machine learning algorithm that is commonly used to solve regression and classification problems.
KNN measures the distance between data to determine classification. There are different methods to measuring the distance, in this case the Euclidean distance was used. The idea is that data that are in the same classification will have a shorter distance from one another, this idea helps the algorithm determine how to classify data in our testing data set. For our experiment, we tried various K and found that the optimal performance of the KNN model was achieved between 4 and 8.

**Bayesian methods -** foundational principles to describe the probability of events. The idea:
- utilize training data to calculate an observed probability of each outcome based on the evidence provided by feature values
- When the classifier is later applied to unlabeled data, it uses the observed probabilities to predict the most likely class for the new features




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Med | 0.78 | 0.98 | 0.87 | 450 |
| Post | 0.58 | 0.09 | 0.15 | 79 |
| Pre | 0.79 | 0.21 | 0.33 | 73 |
| accuracy | | | 0.77 | 602 |
| macro avg | 0.72 | 0.43 | 0.45 | 602 |
| weighted avg | 0.75 | 0.77 | 0.71 | 602 |




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Med | 0.75 | 1.00 | 0.86 | 450 |
| Post | 0.00 | 0.00 | 0.00 | 79 |
| Pre | 0.00 | 0.00 | 0.00 | 73 |
| accuracy | | | 0.75 | 602 |
| macro avg | 0.25 | 0.33 | 0.29 | 602 |
| weighted avg | 0.56 | 0.75 | 0.64 | 602 |

## Mindfulness Intervention for Enhancing Learning Effectiveness

**The purpose of this study includes**:
- Cross-compare learning effects between subjects with different aptitudes, including gender and age.
- Examine the effect of mindfulness meditation on learning.

**Targeted population**: UHD student volunteers, and teachers.

## Data Collection Procedures

*Learning procedure used in the experiment*:

The learning procedure is carried-out on computer. The subject logon to BrainGymmer.com account and choose a brain training game to play. The score of the game is recorded as the subject's performance index.

*Pre-intervention activity*:

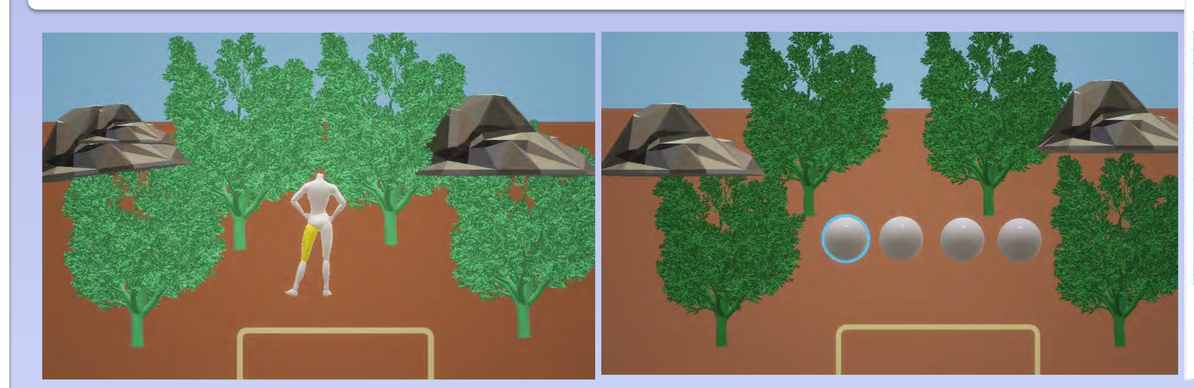
Each volunteer plays the BrainGymmer game as depicted above. The time taken in this step depends on how much time the volunteer uses to finish the game.

*Meditation*:

Each volunteer plays the VR TRIPP meditation game. The TRIPP demo is used to guide the meditation. The entire process take about 10 minutes.

*Post-intervention activities*:

Each volunteer plays the same BrainGymmer game as in the pre-intervention phase. The time taken in this step depends on how much time the volunteer uses to finish the game.



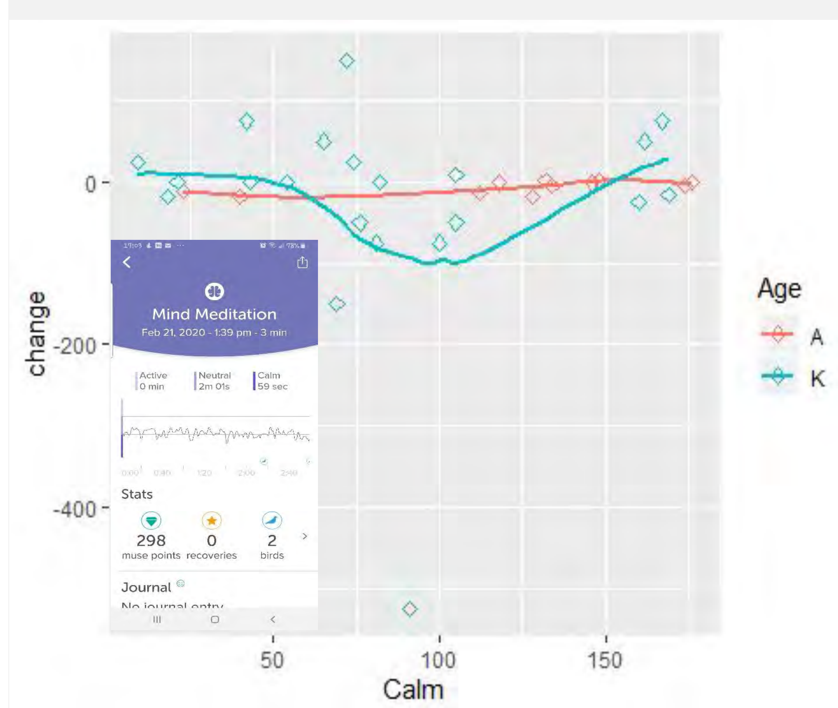

## Data

The dataset has 11 variables.

1. Gender: M or F
2. Age: K (Kid) or A (Adult)
3. Pre-Time: Total puzzle solving time in seconds in the pre-meditation learning session.
4. Pre-Count: Total number of words found when reaching the 5 minutes limit.
5. Active: Number of seconds in "Active" state.
6. Neutral: Number of seconds in "Neutral" state.
7. Calm: Number of seconds in "Calm" state.
8. Recoveries: Number of times the Muse app loses connection to the Muse headband.
9. Birds: Number of bird chirps heard. Every time the subject meditate well, there will be a bid chirp.
10. Post-Time: Total puzzle solving time in seconds in the post-meditation learning session.
11. Post-Counts: Total number of words found when reaching the 5 minutes limit.

## Analysis and Findings

**Hypothesis:**
- Variable Change has negative correlation with variable Calm, which indicates that the more meditation time leads to higher learning rate.




**Findings**:
- Adult group shows slightly positive correlation between variables Change and Calm when the value of Calm is less than 150 seconds; and slightly negative correlation when the value of Calm is above 150 seconds (see the scatterplot above)
- Kids group shows negative correlation between variables Change and Calm when the value of Calm is less than 100 seconds; and positive correlation when the value of Calm is above 100 seconds.
- Overall, females are more easily to get into calm state than males.
- Adults' performance is less affected by meditation than kids.
- Kids show performance increase after short-term meditation, while adults show performance increase after long –term meditation.
- Applying JRipper rule-based learning, using discretized Change variable as the dependent variable, and the Change variable is discretized into 3 values: Decrease, noChange, and Increase. The purpose is to find out whether there is a systematic difference in the learning performance between male and female subjects. Returned rules indicate that fast and slow learners are less likely to be affected by meditation.

## Ripper Algorithm

- Patchwork of efficient heuristics for rule learning
- Three-step process:
  1. Grow: uses the separate and conquer technique to greedily add conditions to a rule until it perfectly classifies a subset of data or runs out of attributes for splitting.
     - The information gain criterion is used to identify the next splitting attribute
  2. Prune: the information gain criterion is used to identify the next splitting attribute.
     - When increasing a rule's specificity no longer reduces entropy, the rule is immediately pruned.
     - Steps one and two are repeated until it reaches a stopping criterion
  3. Optimize

Ripper Rule results for Subject2

Crosstab for "Pre" label

| Predicted Value | FALSE | TRUE |
|---|---|---|
| Test Value | | |
| FALSE | 403 | 0 |
| TRUE | 47 | 52 |

Classification Report for "Pre" label

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| FALSE | 0.9 | 1 | 0.94 | 403 |
| TRUE | 1 | 0.53 | 0.69 | 99 |
| Accuracy | | | 0.91 | 502 |
| Macro avg | 0.95 | 0.76 | 0.82 | 502 |
| Weighted avg | 0.92 | 0.91 | 0.89 | 502 |

Crosstab for "Med" label

| Predicted Value | FALSE | TRUE |
|---|---|---|
| Test Value | | |
| FALSE | 191 | 0 |
| TRUE | 96 | 215 |

Classification Report for "Med" label

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| FALSE | 0.67 | 1 | 0.8 | 191 |
| TRUE | 1 | 0.69 | 0.82 | 311 |
| Accuracy | | | 0.81 | 502 |
| Macro avg | 0.83 | 0.85 | 0.81 | 502 |
| Weighted avg | 0.87 | 0.81 | 0.81 | 502 |

Crosstab for "Post" label

| Predicted Value | FALSE | TRUE |
|---|---|---|
| Test Value | | |
| FALSE | 410 | 0 |
| TRUE | 47 | 45 |

Classification Report for "Post" label

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| FALSE | 0.9 | 1 | 0.95 | 410 |
| TRUE | 1 | 0.49 | 0.66 | 92 |
| Accuracy | | | 0.91 | 502 |
| Macro avg | 0.95 | 0.74 | 0.8 | 502 |
| Weighted avg | 0.92 | 0.91 | 0.89 | 502 |

Ripper Rule results for All Subjects

Crosstab for "Pre" label

| Predicted Value | FALSE | TRUE |
|---|---|---|
| Test Value | | |
| FALSE | 4031 | 5 |
| TRUE | 451 | 101 |

Classification Report for "Pre" label

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| FALSE | 0.9 | 1 | 0.95 | 4036 |
| TRUE | 0.95 | 0.18 | 0.31 | 552 |
| Accuracy | | | 0.9 | 4588 |
| Macro avg | 0.93 | 0.59 | 0.63 | 4588 |
| Weighted avg | 0.91 | 0.9 | 0.87 | 4588 |

Crosstab for "Med" label

| Predicted Value | FALSE | TRUE |
|---|---|---|
| Test Value | | |
| FALSE | 1089 | 0 |
| TRUE | 2195 | 1304 |

Classification Report for "Med" label

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| FALSE | 0.33 | 1 | 0.5 | 1089 |
| TRUE | 1 | 0.37 | 0.54 | 3499 |
| Accuracy | | | 0.52 | 4588 |
| Macro avg | 0.67 | 0.69 | 0.52 | 4588 |
| Weighted avg | 0.84 | 0.52 | 0.53 | 4588 |

Crosstab for "Post" label

| Predicted Value | FALSE | TRUE |
|---|---|---|
| Test Value | | |
| FALSE | 4051 | 0 |
| TRUE | 418 | 119 |

Classification Report for "Post" label

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| FALSE | 0.91 | 1 | 0.95 | 4051 |
| TRUE | 1 | 0.22 | 0.36 | 537 |
| Accuracy | | | 0.91 | 4588 |
| Macro avg | 0.95 | 0.61 | 0.66 | 4588 |
| Weighted avg | 0.92 | 0.91 | 0.88 | 4588 |

## Decision Tree and Random Forest

**A decision tree** is like going to a doctor, who series of questions to determine the cause of your symptoms. We can use a process to create a decision tree and have a series of questions to predict a target class. The advantages of this model include support for non-numeric data, little data preparation, support for dealing with non-linear relationship. The default algorithm used for creation is classification and regression tree (**CART**); it uses the ***Ginny*** impurity or Index measure to construct decisions. This is done by looping over the features and finding the values that gives the lowest probability of misclassifying.
RunTime Efficiency: Loop over all m features, and sort all n samples, O(mn log(n))

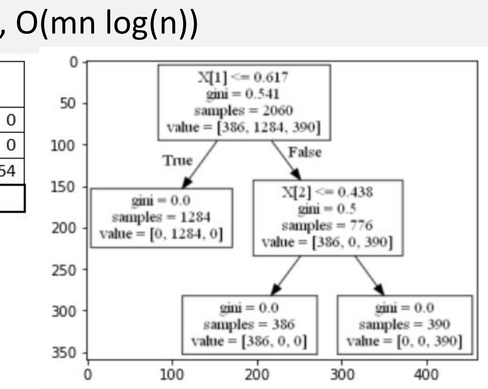
| | Me d | Post | Pre |
|---|---|---|---|
| Med | 575 | 0 | 3 |
| Post | 1 | 3866 | 0 |
| Pre | 0 | 0 | 570 |
| | Fold 1 Accuracy: 99.70% | | |

| | Med | Post | Pre |
|---|---|---|---|
| Med | 571 | 1 | 1 |
| Post | 0 | 3885 | 1 |
| Pre | 0 | 1 | 555 |
| | Fold 1 Accuracy: 99.70% | | |

| | Med | Post | Pre |
|---|---|---|---|
| Med | 549 | 0 | 0 |
| Post | 0 | 3909 | 0 |
| Pre | 2 | 1 | 554 |
| | Fold 1 Accuracy: 99.80% | | |

| | Me d | Post | Pre |
|---|---|---|---|
| Med | 576 | 2 | 2 |
| Post | 1 | 3940 | 2 |
| Pre | 1 | 0 | 491 |
| | Fold 1 Accuracy: 99.90% | | |

| | Med | Post | Pre |
|---|---|---|---|
| Med | 535 | 0 | 0 |
| Post | 1 | 3937 | 0 |
| Pre | 0 | 0 | 541 |
| | Fold 1 Accuracy: 99.70% | | |



**A Random-Forest** is ensemble of decision-trees. It uses bagging to correct the tendency of decision-trees to overfit. By creating many trees, trained on random subsamples of the samples, and random features of data, the variance is lowered. Because they train on subsamples of the data, Random-Forest can evaluate OOB errors and evaluate performance. They can also track feature importance by averaging the feature importance over all the trees. The idea of Random-Forest is to create a **"forest"** of decision trees trained on different column of the training data. If each tree has better than 50% chance of correct classification, you should use its prediction. The Random-Forest is excellent tool for both classification and regression; though it may not be appropriate for gradient-booster trees.
RunTime Efficiency: loop over m features and sort all n samples O(mn log n); Prediction walk the tree O(height)

| | Med | Post | Pre |
|---|---|---|---|
| Med | 578 | 0 | 0 |
| Post | 0 | 3867 | 0 |
| Pre | 0 | 0 | 570 |

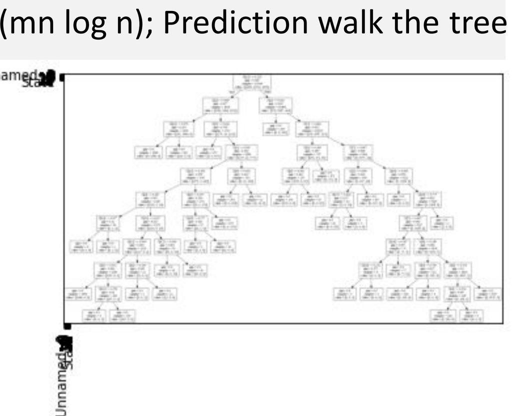
| | Med | Post | Pre |
|---|---|---|---|
| Med | 572 | 1 | 0 |
| Post | 0 | 3886 | 0 |
| Pre | 0 | 0 | 556 |

| | Med | Post | Pre |
|---|---|---|---|
| Med | 549 | 0 | 0 |
| Post | 0 | 3909 | 0 |
| Pre | 1 | 0 | 556 |

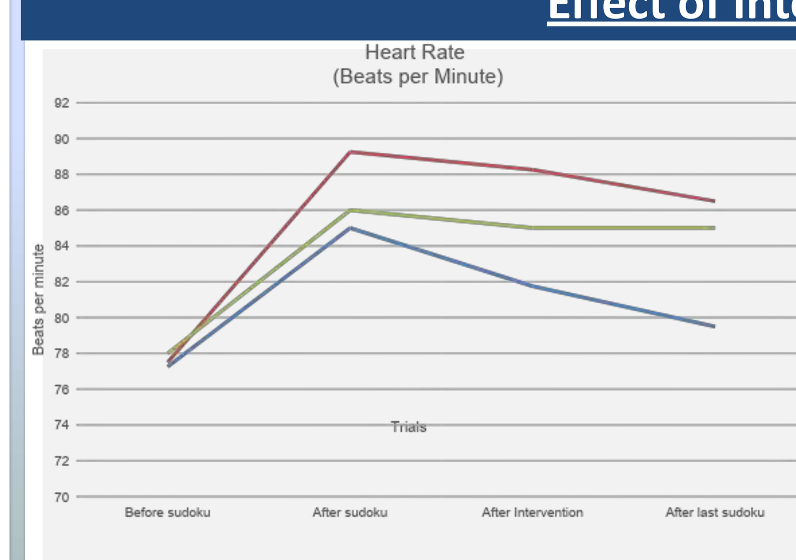
| | Med | Post | Pre |
|---|---|---|---|
| Med | 579 | 1 | 0 |
| Post | 1 | 3942 | 0 |
| Pre | 1 | 0 | 491 |

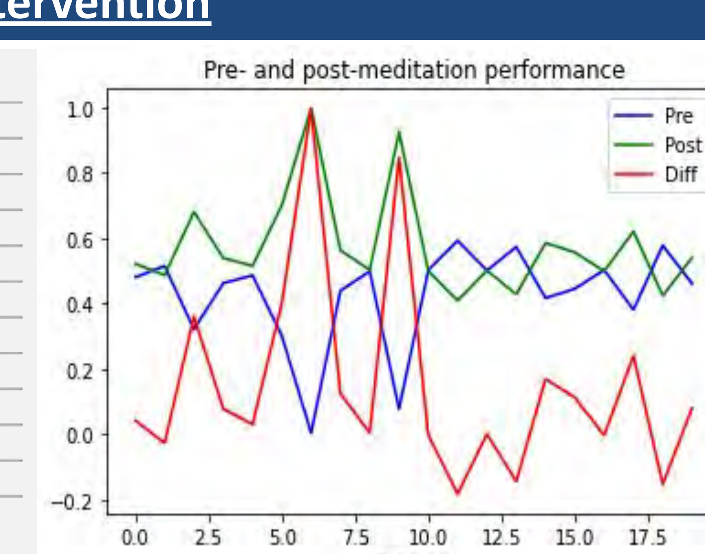
| | Med | Post | Pre |
|---|---|---|---|
| Med | 535 | 0 | 0 |
| Post | 0 | 3938 | 0 |
| Pre | 0 | 0 | 541 |

```
Fold 1 Accuracy: 100.00%
Fold 2 Accuracy: 99.80%
Fold 3 Accuracy: 100.00%
Fold 4 Accuracy: 100.00%
Fold 5 Accuracy: 100.00%
Baseline: 99.96% (0.08%)
```



## Effect of Intervention




## Acknowledgment

This work is supported by the Near Northside Community Service Project and the Service Learning project from the Center for Community Engagement and Service Learning, the University of Houston-Downtown.