CSE 572: Data Mining

Assignment 2 and 3

"Activity Recognition and User-Based Analysis"

[1]Parth Doshi, (ID: 1215200012)
[2]Amogh Bhabal, (ID:1215139822)
[3]Anik Pait (ID:1211225753)
[4]Hardik Shah, (ID:1215487117)
[5]Jay Vaghasiya, (ID:1214287866)

Abstract: As the data is humungous in today's world and keeps on increasing exponentially day by day, our project team has dealt with data from our daily life activities such as eating and non-eating using Myo sensors and the assignment showcases the training and testing of each and every users data on the classification algorithms such as Decision trees, SVM and Neural Networks after dimensionality reduction using PCA.

Keywords: Data, PCA, SVM, Decision trees, Neural Networks

1. Introduction

In this assignment, we need to apply 3 machines,
a) Decision Trees,
b) Support Vector Machines,
c) Neural Networks
to train data of each user for distinguishing eating and non-eating activities.
For Assignment 2, we need to extract eating and non-eating set for each user based on the range of time given in the file. It uses the same features, which were extracted in Assignment 1:

- Mean
- Standard deviation
- Correlation
- Covariance
- Root mean square

and apply Principal Component Analysis for dimensionality reduction of the feature matrix, to get a new feature matrix for each user and split the data into ratios of either 60/40 or 80/20 percent. The majority (higher percentage) of data is extracted to train the model and the remaining percentage of data is used for testing the model. After the testing, the task is to check accuracy metrics such that Precision, Recall, F1 score and accuracy to determine the best model. We intend to repeat the steps above for all users' data and report all accuracy matrices for all the 3-machinery model.

Similarly, for Assignment 3, we need to follow the same procedure used in Assignment 2 (performed above) with a change that we need to combine data of all the given users before passing to the model. In this way, we performed a User Dependent Analysis in Assignment 2 and a User Independent Analysis in Assignment 3.

The Data format is as follows:

IMU file consists of 11 columns: UNIX timestamp, Orientation X, Orientation Y, Orientation Z, Orientation W, Accelerometer X, Accelerometer Y, Accelerometer Z, Gyroscope X, Gyroscope Y, and Gyroscope Z. EMG file consists of 9 columns: UNIX timestamp, EMG 1, EMG 2, EMG 3, EMG 4, EMG 5, EMG 6, EMG 7, and EMG 8.

Reading the Data files:

```
M1 = dlmread(fullfile(forkfileground),',');
```

N1 = dlmread(fullfile(forkfileIMU),',');

O1 = dlmread(fullfile(forkfileEMG),',');

M2 = dlmread(fullfile(spoonfileground),',');

N2 = dlmread(fullfile(spoonfileIMU),',');

O2 = dlmread(fullfile(spoonfileEMG),',');

We have renamed the files as follows:

groundTruth – 0.txt;

IMU file – 1.txt;

EMG file – 2.txt; for ease of access.

2. Model Metrics

2.1 *Precision, Recall, F1 and Accuracy*

To understand Precision, Recall, F1 and Accuracy, we need to first understand what confusion matrix is.
Below is an example of a Confusion matrix:

	PREDICTED CLASS		
ACTUAL CLASS		CLASS = YES	CLASS = NO
	CLASS = YES	A (TP)	B (FN)
	CLASS = NO	C (FP)	D (TN)

HERE,

- TP: TRUE POSITIVE
- FN: FALSE NEGATIVE
- FP: FALSE POSITIVE
- TN: TRUE NEGATIVE

Below are the details what each of the above means:

a). **TP (True Positive)**: It gives the number of classes that were classified as YES and were predicted YES which means it has been correctly predicted by the model.

b). **FN (False Negative):** It gives the number of classes that were classified as YES and were predicted NO which means it has been not been correctly predicted by the model.

c). **FP (False Positive)**: It gives the number of classes that were classified as NO and were predicted YES which means it has been not been correctly predicted by the model.

d). **TN (True Negative):** It gives the number of classes that were classified as NO and were predicted NO which means it has been correctly predicted by the model.

Based on these parameters we can calculate below rates:

a. **True Positive Rate (TPR) or sensitivity:** It is defined as the fraction of positive examples predicted correctly by the model:

$$TPR = TP / (TP + FN)$$

b. **True Negative Rate (TNR) or specificity:** It is defined as the fraction of negative examples predicted correctly by the model:

$$TNR = TN / (TN + FP)$$

c. **False Positive Rate (FPR)**: It is the fraction of negative examples predicted as a positive class:

$$FPR = FP / (FP + TN)$$

d. **False Negative Rate (FNR):** It is the fraction of positive examples predicted as a negative class:

$$FNR = FN / (FN + TP)$$

Based on TP, FN, FP, and TN, we can calculate Precision, Recall, F1 and Accuracy and it is explained below:

a). **Precision**: Precision computes the fraction of records that return accurate results from a pool of relevant data. Higher precision implies a lower number of false positive errors committed by the classifier:

$$P = TP / (TP + FP)$$

b). **Recall**: Recall computes the fraction of samples returned from the pool of relevant samples available. Higher recall implies very few positive examples are misclassified as the negative class. The recall is equivalent to the true positive rate (TPR):

$$R = TP / (TP + FN)$$

c). **F1 measure**: Precision and Recall can be summarized into another metric, called the F1 Measure, which is the harmonic mean of precision and recall:

$$F1 = 2RP / (R + P)$$

d). **Accuracy**: It is the ratio of all classes that were correctly classified to the total number of class:

$$A = (TP + TN) / (TP + TN + FP + FN)$$

3. Assumptions

- The data collection application does not guarantee what the start time between Myo data and Video frame are the same. The end time between Myo and Video, however, is the approximately same.
- While running the code file 'Assignment_2.m' and 'Assignment_3.m', the user data file (all the files) belongs to the same folder as the MATLAB files.

4. Results

For Assignment 2:

1. **Support Vector Machine (SVM)**

Support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. In our assignment, we have used MATLAB function ***fitcsvm*** to classify eating and non-eating classes.

Below is the code snippet:

```
trainLabels=[K1(1:split,n1+1);K2(1:split,n2+1)];
testLabels=[K1(split:m1,n1+1);K2(split:m2,n2+1
)];
Mdl = fitcsvm(training,trainLabels);
label = predict(Mdl,test);
```

SUPPORT VECTOR MACHINE RESULTS				
USERS	***ACCURACY***	***PRECISION***	***RECALL***	***F1 SCORE***
USER9	1	1	1	1
USER10	0.78788	0.75676	0.84848	0.8
USER11	0.98485	1	0.9697	0.98462

USER12	0.43939	0.4	0.24242	0.30189
USER13	0.95833	1	0.91667	0.95652
USER14	0.54545	1	0.090909	0.16667
USER16	0.92424	0.86842	1	0.92958
USER17	0.93939	1	0.87879	0.93548
USER18	0.93548	0.93548	0.93548	0.93548
USER19	0.86364	0.81579	0.93939	0.87324
USER21	0.90909	1	0.81818	0.9
USER22	1	1	1	1
USER23	0.63636	0.64516	0.60606	0.625
USER24	0.98485	1	0.9697	0.98462
USER25	0.33333	0.34921	0.81481	0.48889
USER26	0.78788	0.80645	0.75758	0.78125
USER27	0.9697	0.94286	1	0.97059
USER28	0.95238	0.94118	0.9697	0.95522
USER29	0.78788	0.71111	0.9697	0.82051
USER30	0.90909	0.90909	0.90909	0.90909
USER31	0.9697	0.94286	1	0.97059
USER32	0.86	0.85714	0.88889	0.87273
USER33	0.71212	0.76923	0.60606	0.67797
USER34	0.71212	1	0.42424	0.59574
USER36	0.80303	0.76316	0.87879	0.8169
USER37	0.74242	0.76667	0.69697	0.73016
USER38	0.95455	0.96875	0.93939	0.95385
USER39	0.86364	0.92857	0.78788	0.85246
USER40	0.62121	0.63333	0.57576	0.60317
USER41	0.98485	0.97059	1	0.98507
AVERAGE	0.82320	0.85606	0.81449	0.81258

2. **Decision Tree**

Decision tree learning is a method commonly used in data mining. The goal is to create a model that predicts the value of a target variable based on several input variables. Each interior node corresponds to one of the input variables; there are edges to children for each of the possible values of that input variable. Each leaf represents a value of the target variable given the values of the input variables represented by the path from the root to the leaf. In our assignment, we have used MATLAB function **fitctree** to classify eating and non-eating classes.

Below is the code snippet:

```
trainLabels=[K1(1:split,n1+1);K2(1:split,n2+1)];
testLabels=[K1(split:m1,n1+1);K2(split:m2,n2+1
)];
tree=fitctree(training,trainLabels);
label=predict(tree,test);
```

DECISION TREE RESULTS				
USERS	***ACCURACY***	***PRECISION***	***RECALL***	***F1 SCORE***
USER9	1	1	1	1
USER10	0.80303	0.76316	0.87879	0.8169
USER11	0.95455	1	0.90909	0.95238
USER12	0.90909	0.88571	0.93939	0.91176
USER13	0.97917	0.96	1	0.97959
USER14	1	1	1	1
USER16	0.95455	0.91667	1	0.95652
USER17	0.87879	0.90323	0.84848	0.875
USER18	0.83871	0.8	0.90323	0.84848
USER19	0.95455	1	0.90909	0.95238
USER21	0.90909	1	0.81818	0.9
USER22	0.9697	1	0.93939	0.96875
USER23	0.87879	0.87879	0.87879	0.87879
USER24	0.90909	0.90909	0.90909	0.90909
USER25	0.63768	0.53571	0.55556	0.54545
USER26	0.98485	1	0.9697	0.98462
USER27	0.98485	0.97059	1	0.98507
USER28	1	1	1	1
USER29	0.77273	0.725	0.87879	0.79452
USER30	0.92424	0.88889	0.9697	0.92754
USER31	0.89394	0.84211	0.9697	0.90141
USER32	0.92	0.87097	1	0.93103
USER33	0.71212	0.71875	0.69697	0.70769
USER34	0.80303	0.88462	0.69697	0.77966
USER36	0.83333	0.78947	0.90909	0.84507
USER37	0.77273	0.875	0.63636	0.73684
USER38	0.90909	1	0.81818	0.9
USER39	0.83333	0.84375	0.81818	0.83077
USER40	0.69697	0.65116	0.84848	0.73684
USER41	0.95455	0.94118	0.9697	0.95522
AVERAGE	0.88575	0.88513	0.88903	0.88371

3. **Neural Networks**

An ANN is a model based on a collection of connected units or nodes called "artificial neurons", which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit information, a "signal", from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it. In common ANN implementations, the signal at a connection between artificial neurons is a real number, and the output of each artificial neuron is computed

by some non-linear function of the sum of its inputs. The connections between artificial neurons are called "edges". Artificial neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Artificial neurons may have a threshold such that the signal is only sent if the aggregate signal crosses that threshold. Typically, artificial neurons are aggregated into layers. Different layers may perform different kinds of transformations on their inputs. Signals travel from the first layer (the input layer) to the last layer (the output layer), possibly after traversing the 10 hidden layers. We have 5 input nodes denoting 5 data features and 2 output nodes denoting labels 0 (non-eating) and 1 (eating). In our Assignment, we have used MATLAB tool **patternnet** to classify eating and non-eating classes.

Below is the code snippet:

```
net = patternnet(10);
net.trainParam.showWindow=0;
net = train(net,trainnet,tt);
tsn=net(testnet);
```

NEURAL NETWORKS OUTPUT				
USERS	***ACCURACY***	***PRECISION***	***RECALL***	***F1 SCORE***
USER9	1	1	1	1
USER10	0.86364	0.83333	0.90909	0.86957
USER11	0.90909	1	0.81818	0.9
USER12	0.86364	0.81579	0.93939	0.87324
USER13	0.95833	0.95833	0.95833	0.95833
USER14	1	1	1	1
USER16	0.93939	0.89189	1	0.94286
USER17	0.9697	1	0.93939	0.96875
USER18	0.85484	0.78947	0.96774	0.86957
USER19	0.98485	0.97059	1	0.98507
USER21	0.92424	0.96667	0.87879	0.92063
USER22	0.98485	1	0.9697	0.98462
USER23	0.83333	0.86667	0.78788	0.8254
USER24	0.98485	1	0.9697	0.98462
USER25	0.6087	0.5	0.2963	0.37209
USER26	0.90909	0.96552	0.84848	0.90323
USER27	0.98485	0.97059	1	0.98507
USER28	0.93651	0.91429	0.9697	0.94118
USER29	0.80303	0.72727	0.9697	0.83117
USER30	0.95455	0.96875	0.93939	0.95385
USER31	1	1	1	1
USER32	0.86	0.85714	0.88889	0.87273
USER33	0.84848	0.82857	0.87879	0.85294
USER34	0.92424	0.96667	0.87879	0.92063

USER36	0.81818	0.76923	0.90909	0.83333
USER37	0.75758	0.86957	0.60606	0.71429
USER38	0.98485	1	0.9697	0.98462
USER39	0.87879	0.93103	0.81818	0.87097
USER40	0.74242	0.72222	0.78788	0.75362
USER41	0.95455	0.91667	1	0.95652
AVERAGE	0.90122	0.90001	0.89664	0.89430

For Assignment 3:

For assignment 3, we must do user independent analysis, that is we have combined the fork and spoon data for all the users and trained our previous models (as mentioned in Assignment 2) on the data. All the other approaches for training and testing the model is exactly like what we did in Assignment 2.

RESULTS OF SVM:

SVM ACCURACY	*SVM PRECISION*	*SVM RECALL*	*SVM F1 SCORE*
0.475987193	0.415224913	0.128617363	0.196399345

RESULTS OF DECISION TREES:

DECISION TREE ACCURACY	*DECISION TREE PRECISION*	*DECISION TREE RECALL*	*DECISION TREE F1 SCORE*
0.845784418	0.813229572	0.896034298	0.852626211

RESULTS OF NEURAL NETWORK:

NEURAL NETWORK ACCURACY	*NEURAL NETWORK PRECISION*	*NEURAL NETWORK RECALL*	*NEURAL NETWORK F1 SCORE*
0.740661686	0.719332679	0.785637728	0.75102459

5. Conclusion

We implemented the following classification algorithms:

- Support Vector Machine (SVM),
- Decision Tree and
- Neural Network

for both User dependent data in our Assignment 2 and User-independent data in our Assignment 3. The results for each model for each user have been presented in the above sections and

similarly, the results have been presented in the above sections for all the users for the second assignment as well for the third assignment. The metrics such as Precision, Recall, and the F1 score has been presented in both the cases and for each algorithm. The averages of the metrics for user-independent analysis has also been presented, for comparison to the metrics of the user independent analysis.

From the performance metrics results, we see that the user dependent analysis gives better accuracies than the user independent analysis. But user independent analysis is more relevant, even though the SVM classifier is not suited for it.

Acknowledgment:

We would like to thank Professor Dr. Ayan Banerjee for his constant guidance and support over our questions and our TA, Bernard for his prompt response to our queries.