Date:11/28/17

CSE 572 Data Mining Assignment 4

At

Arizona State University

By

Bharath Kumar Suresh (1211182086)
Dhanajayan Santhankrishnan (1211181423)
Joel Jery Mascarenhas (1211194319)
Vamsi Krishna Godavarthi (1210933825)

Date:11/28/17

Definitions

Decision tree: A decision support tool which uses a tree-like model of decisions and their possible outcomes.[1] In this phase, we have utilized the **fitctree** provided by Matlab. As per Matlab fitctree is “fit binary classification decision tree for multiclass classification”[2]

Support Vector Machine: Support Vector Machines(SVM) are supervised learning models with association learning algorithms that analyze data used for classification and regression analysis.[3] In this phase, we have utilized the fitcsvm provided by Matlab. As per Matlab fitcsvm is “fitcsvm trains or cross-validates a support vector machine (SVM) model for two-class (binary) classification on a low- through moderate-dimensional predictor data set. fitcsvm supports mapping the predictor data using kernel functions, and supports SMO, ISDA, or *L*1 soft-margin minimization via quadratic programming for objective-function minimization.”[4]
We have two different types of SVMs, one class is linear SVM, which separates the points in the space into two categories which are as wide from each other as possible. The other class is the non-linear SVM, which maps the inputs into high dimensional feature space and can classify the points in that.

Artificial Neural Network: Computing systems inspired by the biological neural networks like brain networks. Artificial Neural Network is a collection of artificial neurons, where each neuron can transmit the signal to other connected neurons, the receiving neurons can process the signal and send signals further. In most cases, the signal from a neuron is mostly a number and the output of each neuron either increases/decreases the strength of that real number.[5]
In this phase, we have utilized the Neural Network Toolbox functions provided by Matlab. It helps us in create, train, and simulate shallow and deep learning neural networks. [6]

True positives(TP): These refer to the positive tuples that were correctly labeled by the classifier.[7]

True negatives(TN): These are the negative tuples that were correctly labeled by the classifier.[7]

False positives(FP): These are the negative tuples that were incorrectly labeled as positive.[7]

False negatives(FN): These are the positive tuples that were mislabeled as negative.[7]

Precision(p): It computes the fraction of records that actually turn out to be positive in the group the classifier has declared as positive class. Higher precision implies a lower number of false positive errors committed by the classifier.[8]

Date:11/28/17

$$p = \frac{TP}{TP + FP}$$

Recall(r): Recall computes the fraction of positive examples correctly predicted by the classifier. Higher recall implies very few positive examples are misclassified as the negative class. Recall is equivalent to the true positive rate (TPR).[8]

$$r = \frac{TP}{TP + FN}$$

F1 Measure (f1): Precision and Recall can be summarized into another metric, called the F_1 Measure, which is the harmonic mean of precision and recall.The harmonic mean of two numbers tends to be closer to the smaller of the two numbers.Therefore, a high value of the F_1 measure ensures that both precision and recall are high.[8]

$$F_1 = \frac{2}{\frac{1}{r} + \frac{1}{p}} = \frac{2rp}{r + p}$$

ROC: The Receiver Operating Characteristic (ROC) plots the true positive rate on the y-axis against the false positive rate on the x-axis.Each point on the curve corresponds to one of the models induced by the classifier.[8]

AUC: Area under the curve(AUC) is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming 'positive' ranks higher than 'negative').[9]

Date:11/28/17

Recap of Phase 3:

List of features selected from the sensor data to differentiate eating vs non-eating

Sensor	Feature
Gyroscope X	Normalized Wilson Amplitude
Gyroscope Y	Normalized Wilson Amplitude
Gyroscope Z	Normalized Wilson Amplitude
Orientation X	Variance
Orientation Y	Variance
Orientation Z	Variance
Orientation W	Variance

After performing PCA, we selected the first three Principal Components which had the highest eigenvalues.

The eigenvectors corresponding to the first 3 eigenvalues are as follows (column-wise):

First Principal Component	Second Principal Component	Third Principal Component
0.589141782	-0.394494431	-0.705084822
0.53415911	0.844910609	-0.026226028
0.606046708	-0.361242092	0.708628295
-0.003631418	-0.001444113	0.001402396
-0.012836194	-0.00129374	0.002906064
-0.010464874	-0.000227801	0.001192709
-0.002434139	-0.000738995	0.00130147

Date:11/28/17

Phase 4 Implementation:

Type of Models selected in Implementation:

Decision Tree (fitctree): For this phase the default function provided by matlab was used. Matlab by default sets PredictorSelection = allsplits (by default), and we use this version of Decision tree to classify the data. Once the training is completed, the test data is loaded and it returns the predicted label and scores (probability), which are used for the computation of the accuracy metrics.

Support Vector Machine (fitcsvm): We experimented with various kernels such as linear, RBF and polynomial kernels. Linear kernel didn't perform well for any accuracy measures used (i.e, Precision, Recall, F1 and AUC). Clearly eating and non-eating actions are not linearly separable. Although, RBF kernel performed well for Precision we found out that it didn't perform well for Recall and hence F1. We found out that a polynomial kernel with degree = 12 gave us the best results in terms of all the measures we calculate. Hence, we use SVM with a polynomial kernel with degree = 12.

Neural Network (neural network toolbox): We decided to move forward with the feedforwardnet implementation provided by Matlab. The network is created using the feedforwardnet() function and the number of hidden layers is set to 10. Patternnet() and cascadeforwardnet() are used for pattern recognition related applications and hence decided not to experiment with them. To train the data, we use the randomly sampled input also pass the train-data labels (the transpose matrices are used for training). For the test data, we use the net() function and that gives the predicted output, as to which class the particular test data belongs to. We set the metric that if the predicted score is less than or equal to 0.5, it is non eating action and for values greater than 0.5 and the maximum possible value being 1, the test data would belong to eating action class. For reporting the accuracy metrics, we compute the score for the other class by doing a (1 - score) conversion and report the data.

User Dependant Analysis:

For each group's spoon and fork data, we have combined all the data for both the categories and have randomly selected 60% (random sampling) of the data as train data and the rest was used as test data to report the accuracy metrics such as Precision, Recall, F1 score and ROC for each of the three models, which are Decision Trees (fitctree) and SVM (fitcsvm) and Neural Network (Neural Network Toolbox)

User Independent Analysis:

We selected the first 10 groups data as the train data, in which we combined the data for both the categories of spoon and fork. The data of the other 23 groups were selected without combining spoon and fork data and used to test the models and report the accuracy metrics.

Date:11/28/17

Task 1: Results (User Dependent Analysis)

	Decision Tree				SVM				Neural Network			
Group No.	Precision	Recall	F1	AUC	Precision	Recall	F1	AUC	Precision	Recall	F1	AUC
1	0.65625	0.617647	0.636364	0.585294	1	0.264706	0.418605	0.7	0.9	0.529412	0.666667	0.732353
2	0.925926	0.925926	0.925926	0.945419	0.96	0.888889	0.923077	0.97271	0.952381	0.740741	0.833333	0.892788
3	0.846154	0.88	0.862745	0.888718	1	0.72	0.837209	0.869744	0.807692	0.84	0.823529	0.895385
4	0.8	0.965517	0.875	0.914286	0.952381	0.689655	0.8	0.944828	0.857143	0.62069	0.72	0.858128
5	0.647059	0.6875	0.666667	0.697266	0.740741	0.625	0.677966	0.786133	0.666667	0.75	0.705882	0.782227
6	0.966667	0.966667	0.966667	0.968627	1	0.9	0.947368	0.987255	1	0.833333	0.909091	0.978431
7	0.689655	0.625	0.655738	0.730957	1	0.5	0.666667	0.758789	0.85	0.53125	0.653846	0.773438
8	0.892857	0.78125	0.833333	0.845215	0.954545	0.65625	0.777778	0.888672	0.956522	0.6875	0.8	0.892578
9	0.735294	0.757576	0.746269	0.737048	0.875	0.424242	0.571429	0.856305	1	0.545455	0.705882	0.879765
10	0.703704	0.655172	0.678571	0.759113	0.607143	0.586207	0.596491	0.697537	0.62963	0.586207	0.607143	0.755665
11	0.823529	0.8	0.811594	0.848768	1	0.685714	0.813559	0.912315	0.911765	0.885714	0.898551	0.949754
12	0.764706	0.83871	0.8	0.815054	0.952381	0.645161	0.769231	0.753763	0.8	0.387097	0.521739	0.607527
13	0.576923	0.5	0.535714	0.513235	0.75	0.5	0.6	0.647549	0.846154	0.733333	0.785714	0.894608
14	0.818182	0.818182	0.818182	0.83089	1	0.666667	0.8	0.886608	0.814815	0.666667	0.733333	0.802542
15	0.878788	0.90625	0.892308	0.90918	0.954545	0.65625	0.777778	0.964844	0.9	0.84375	0.870968	0.943359
16	0.740741	0.526316	0.615385	0.637652	0.705882	0.315789	0.436364	0.624494	0.619048	0.342105	0.440678	0.484818
17	1	0.828571	0.90625	0.914286	1	0.714286	0.833333	0.901478	1	0.742857	0.852459	0.933005
18	0.783784	0.878788	0.828571	0.823069	1	0.727273	0.842105	0.936461	0.8	0.848485	0.823529	0.899316
19	0.714286	0.78125	0.746269	0.77002	0.896552	0.8125	0.852459	0.917969	0.794118	0.84375	0.818182	0.892578
20	0.933333	0.903226	0.918033	0.938508	1	0.741935	0.851852	0.967742	0.92	0.741935	0.821429	0.947581
21	0.833333	0.857143	0.84507	0.868473	0.933333	0.8	0.861538	0.889655	0.964286	0.771429	0.857143	0.923153
22	0.782609	0.580645	0.666667	0.818311	0.92	0.741935	0.821429	0.908918	0.888889	0.774194	0.827586	0.885199
23	0.96	0.75	0.842105	0.881836	1	0.6875	0.814815	0.938477	0.958333	0.71875	0.821429	0.932617
24	0.75	0.870968	0.80597	0.892962	0.931034	0.870968	0.9	0.904203	0.84375	0.870968	0.857143	0.893451
25	0.543478	0.862069	0.666667	0.710345	0.678571	0.655172	0.666667	0.780296	0.453125	1	0.623656	0.608867
26	0.8125	0.764706	0.787879	0.811765	0.952381	0.588235	0.727273	0.785294	0.684211	0.764706	0.722222	0.733333

Date:11/28/17

27	0.84	0.807692	0.823529	0.791209	0.947368	0.692308	0.8	0.942308	0.909091	0.769231	0.833333	0.825549
28	0.742857	0.83871	0.787879	0.816716	0.862069	0.806452	0.833333	0.886608	0.714286	0.806452	0.757576	0.831867
29	0.92	0.851852	0.884615	0.887888	1	0.814815	0.897959	0.987988	0.952381	0.740741	0.833333	0.837838
30	0.758621	0.709677	0.733333	0.762903	0.869565	0.645161	0.740741	0.795699	0.666667	0.645161	0.655738	0.774194
31	0.571429	0.347826	0.432432	0.485822	0.619048	0.565217	0.590909	0.672968	0.7	0.608696	0.651163	0.659735
32	0.8	0.888889	0.842105	0.798611	0.931034	0.75	0.830769	0.888889	0.928571	0.722222	0.8125	0.871032
33	0.896552	0.742857	0.8125	0.829557	1	0.6	0.75	0.940887	0.8	0.8	0.8	0.870936
Average	**0.791188**	**0.77323**	**0.777283**	**0.800879**	**0.908896**	**0.664797**	**0.758446**	**0.857496**	**0.833016**	**0.717965**	**0.758933**	**0.831625**

Date:11/28/17

Task 2: Results (User Inependent Analysis)

	Decision Tree				SVM				Neural Network			
Group No.	Precision	Recall	F1	AUC	Precision	Recall	F1	AUC	Precision	Recall	F1	AUC
11	0.767442	0.825	0.795181	0.830625	0.927536	0.8	0.85906	0.951875	0.822222	0.925	0.870588	0.947031
12	0.743243	0.723684	0.733333	0.786011	0.893617	0.552632	0.682927	0.864266	0.75	0.631579	0.685714	0.832064
13	0.618182	0.85	0.715789	0.788438	0.77907	0.8375	0.807229	0.827734	0.636364	0.875	0.736842	0.828203
14	0.74026	0.7125	0.726115	0.755313	0.927273	0.6375	0.755556	0.850156	0.826087	0.7125	0.765101	0.830313
15	0.8875	0.8875	0.8875	0.933594	0.969697	0.8	0.876712	0.970313	0.911392	0.9	0.90566	0.977188
16	0.626667	0.5875	0.606452	0.673438	0.777778	0.4375	0.56	0.661094	0.689655	0.5	0.57971	0.698438
17	0.802469	0.8125	0.807453	0.867813	1	0.6625	0.796992	0.955	1	0.7375	0.848921	0.951875
18	0.654206	0.875	0.748663	0.764297	0.776316	0.7375	0.75641	0.798906	0.683673	0.8375	0.752809	0.840313
19	0.625	0.75	0.681818	0.687109	0.662651	0.6875	0.674847	0.758438	0.674157	0.75	0.710059	0.80125
20	0.684685	0.962025	0.8	0.749239	0.764045	0.860759	0.809524	0.842493	0.768421	0.924051	0.83908	0.91636
21	0.514085	0.9125	0.657658	0.473203	0.577586	0.8375	0.683673	0.771406	0.549618	0.9	0.682464	0.734219
22	0.561404	0.790123	0.65641	0.648453	0.60396	0.753086	0.67033	0.752782	0.586207	0.839506	0.690355	0.765585
23	0.757895	0.9	0.822857	0.823906	0.910256	0.8875	0.898734	0.964844	0.869048	0.9125	0.890244	0.955313
24	0.795181	0.825	0.809816	0.851094	0.850746	0.7125	0.77551	0.889063	0.863014	0.7875	0.823529	0.898281
25	0.568182	0.625	0.595238	0.604297	0.708333	0.6375	0.671053	0.700781	0.666667	0.625	0.645161	0.719688
26	0.663043	0.7625	0.709302	0.726719	0.77027	0.7125	0.74026	0.813281	0.77381	0.8125	0.792683	0.850313
27	0.649485	0.926471	0.763636	0.867539	0.935484	0.852941	0.892308	0.932093	0.935484	0.852941	0.892308	0.935121
28	0.670213	0.7875	0.724138	0.722109	0.7625	0.7625	0.7625	0.841875	0.764706	0.8125	0.787879	0.852813
29	0.778947	0.925	0.845714	0.915781	0.930556	0.8375	0.881579	0.961563	0.934211	0.8875	0.910256	0.959531
30	0.677419	0.828947	0.745562	0.728705	0.662651	0.723684	0.691824	0.784453	0.659341	0.789474	0.718563	0.786877
31	0.619048	0.45614	0.525253	0.599415	0.733333	0.192982	0.305556	0.524777	0.538462	0.245614	0.337349	0.532472
32	0.674157	0.75	0.710059	0.756563	0.830769	0.675	0.744828	0.842344	0.859375	0.6875	0.763889	0.840469
33	0.657658	0.9125	0.764398	0.777188	0.825	0.825	0.825	0.904844	0.742268	0.9	0.813559	0.906563
Average	**0.68419**	**0.799452**	**0.731841**	**0.753515**	**0.807801**	**0.714069**	**0.744453**	**0.833234**	**0.761051**	**0.775898**	**0.758379**	**0.841751**

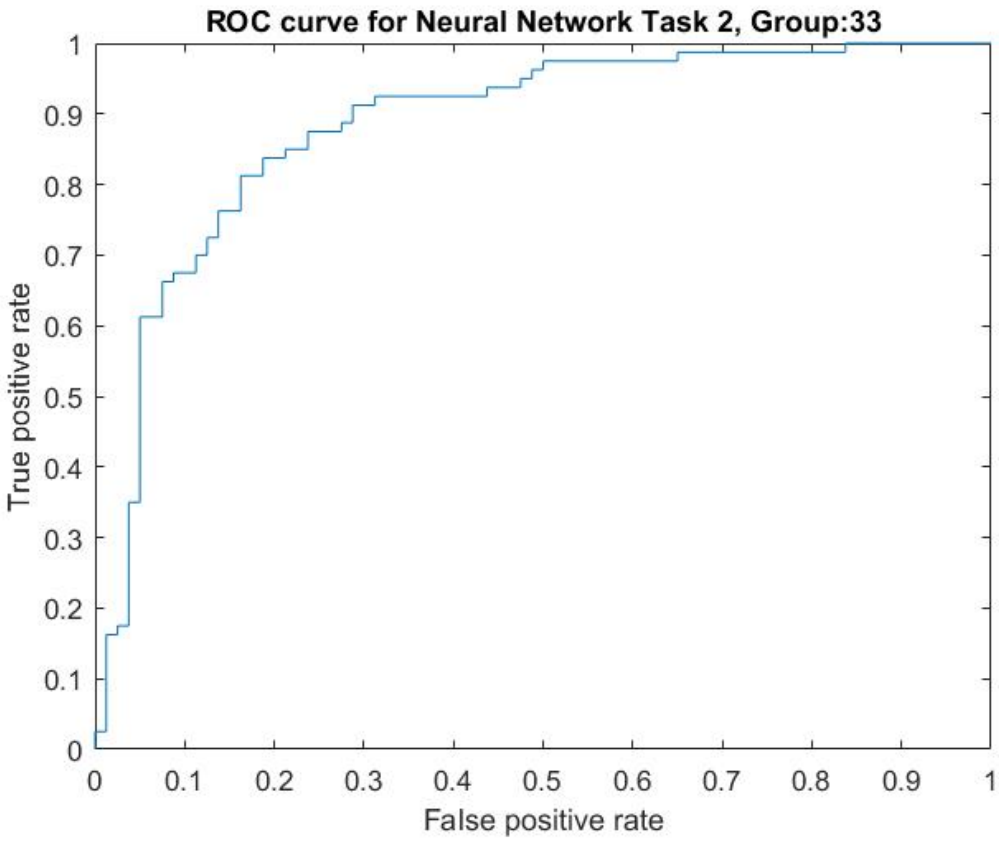
Date:11/28/17

Execution:

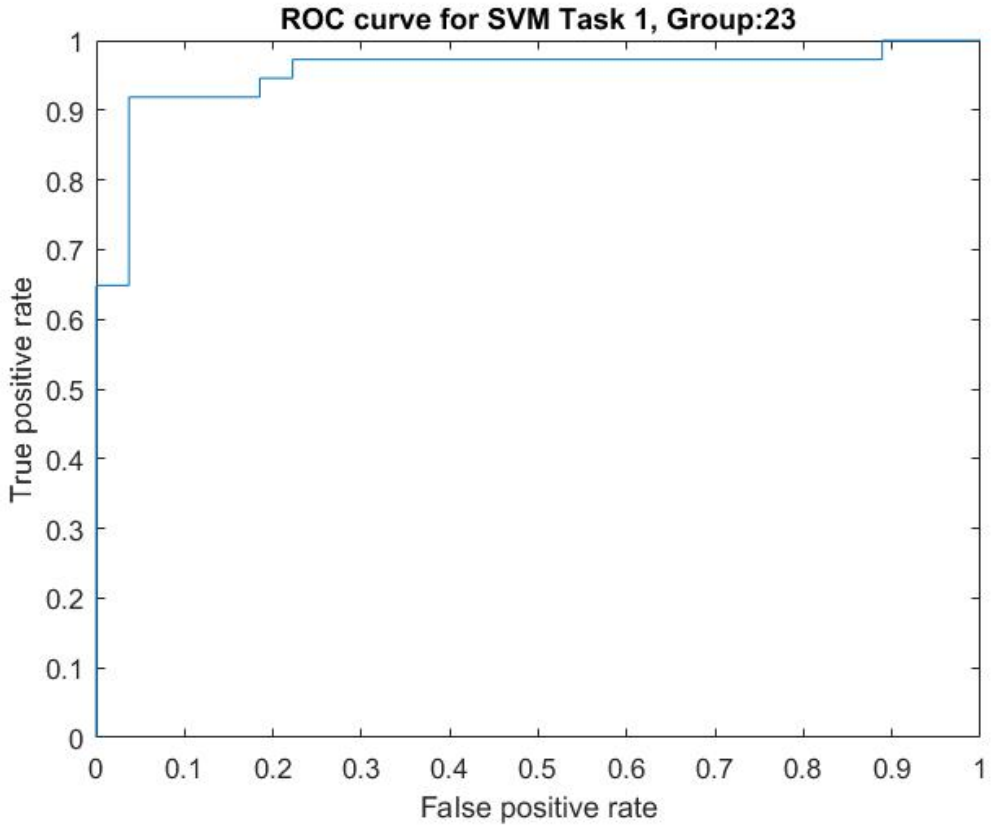
1. Run **Assign4.m**. The results of the first task (user dependent analysis) will be stored in firstVals.mat and second task (user independent analysis) results will be stored in secondVals.mat

ROC curves will be displayed as results are predicted for each group under task 1 and task 2.

Sample ROC graph:



Date:11/28/17



References:

1. Decision tree Wiki: https://en.wikipedia.org/wiki/Decision_tree
2. fitctree Matlab: https://www.mathworks.com/help/stats/fitctree.html
3. Support Vector Machine Wiki: https://en.wikipedia.org/wiki/Support_vector_machine
4. fitcsvm Matlab: https://www.mathworks.com/help/stats/fitcsvm.html
5. Artificial Neural Network Wiki: https://en.wikipedia.org/wiki/Artificial_neural_network
6. Neural Network Toolbox: https://www.mathworks.com/products/neural-network.html
7. Han, Jiawei, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.
8. Tan, Pang-Ning, Michael Steinbach, and Vipin Kumar. "Data mining cluster analysis: basic concepts and algorithms." *Introduction to data mining* (2013).
9. Fawcett, Tom (2006); *An introduction to ROC analysis*, Pattern Recognition Letters, 27, 861–874.