**Github link for sourcecode and evidence:** <https://github.com/priyanka-bharat/CW3>

**Data Preparation:**

The following steps have written for all the datasets in R:

1. Converted Emotion as factor
2. Removed Duplicate Records
3. Performed PCA and generated reduced dataset, the reduced dataset are stored as:

|  |  |  |
| --- | --- | --- |
| Input | Reduced data in csv format | Reduced data in arff format |
| fer2017-training.csv | train.csv | train.arff |
| fer2017-testing.csv | test.csv | test.arff |
| fer2017-training-happy.csv | happy\_train.csv | happy\_train.arff |
| fer2017-testing-happy.csv | happy\_train.csv | happy\_test.arff |
| Train + 3000 instances of test | train3000.csv | train3000.arff |
| Train + 6000 instances of test | train6000.csv | train6000.arff |

**Variation in performance with size of the training and testing set**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DataSet | J48 | User Classifier | Random tree | MultiLayer Perceptron |
| Train | 88.98% | 65.84% | 24.04% | 34.15% |
| Test | 88.54% | 24.90% | 23.56% | 31.45% |
| Train + 3000 Test | 88.80% | 25.81% | 24.52% | 34.1494 % |
| Train + 6000 Test | 88.89% | 25.86% | 25.39% | 34.2236 % |
| Happy-Train | 81.51% | 74.21% | 65.60% |  |
| Happy-Test | 83.07% | 75.09% | 65.84% |  |

The above results are obtained based on the default settings in each algorithm. Each Algorithm behaves in a different way and the results are varying per model.

Observation:

the best model is by using J48 Classifier.

In J48 overfitting is not an issue where as in the user classifier result – as the record increases, the accuracy % decreases.

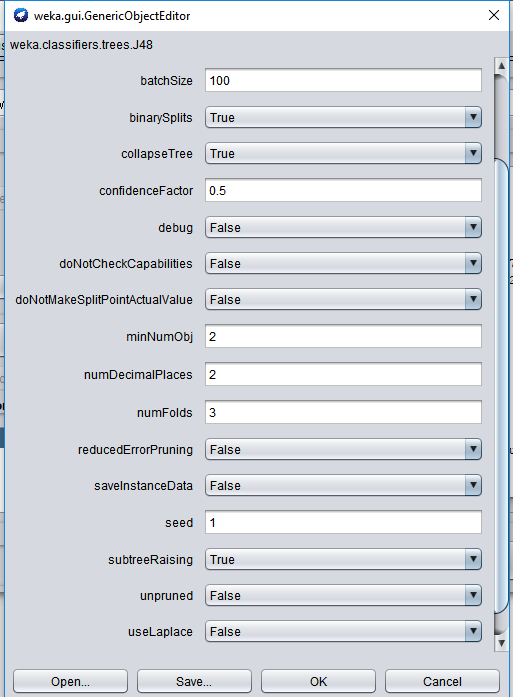
**Variation in performance with change in the learning paradigm**

The maximum time taken to built a model was building the neural network model with learning rate is 0.6 and momentum as 0.5 which is 223.84 seconds. Time taken is as follows:

1. Train Data :157.95
2. Test Data: 198.94
3. Train with 3000 Test: 198.94
4. Train with 6000 Test: 215.69

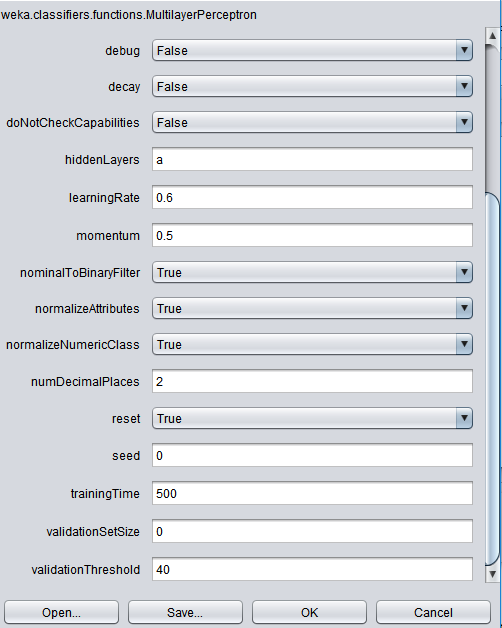
Certainly, the number of time taken to load the data and create the models were reduced after reducing the features using PCA Component. **However there want any difference of accuracy with the full record set and the reduced record set**.

**Variation in performance with varying learning parameters in Decision Trees**



The accuracy % was around 88% with default setting and also by altering the confidenceFactor by 0.25. 0.5 and 0.75. the binary splits were set as True and False and also with the unpruned condition as true and False.

**Variation in performance with varying learning parameters in Neural Nets**



The best results of neural network are when the learning rate is 0.3 and momentum is at 0.25. The validation threshold was also increased to 40.

**Variation in performance according to different metrics**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **J48** | TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area |
| Train | 0.24 | 0.162 | 0.242 | 0.24 | 0.241 | 0.079 | 0.539 | 0.193 |
| Test | 0.236 | 0.16 | 0.236 | 0.236 | 0.236 | 0.076 | 0.538 | 0.189 |
| Train+3000 | 0.252 | 0.16 | 0.254 | 0.252 | 0.252 | 0.092 | 0.553 | 0.205 |
| Train+6000 | 0.256 | 0.159 | 0.258 | 0.256 | 0.257 | 0.097 | 0.556 | 0.206 |

The Result are around the same range for entire dataset and the accuracy % remained same throughout.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **User Classifier** | TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area |
| Train | 0.658 | 0.578 | 0.656 | 0.658 | 0.657 | 0.081 | 0.54 | 0.642 |
| Test | 0.249 | 0.249 | 0.062 | 0.249 | 0.099 | 0 | 0.499 | 0.173 |
| Train+3000 | 0.258 | 0.258 | 0.067 | 0.258 | 0.106 | 0 | 0.5 | 0.177 |
| Train+6000 | 0.259 | 0.259 | 0.067 | 0.259 | 0.106 | 0 | 0.5 | 0.177 |

In User classifier, we have a clear example of the model performance varying with respect to the data size and hence this isn’t a good model for this particular dataset.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Random tree** | TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area |
| Train | 0.24 | 0.162 | 0.242 | 0.24 | 0.241 | 0.079 | 0.539 | 0.193 |
| Test | 0.236 | 0.16 | 0.236 | 0.236 | 0.236 | 0.076 | 0.538 | 0.189 |
| Train+3000 | 0.245 | 0.162 | 0.246 | 0.245 | 0.246 | 0.083 | 0.541 | 0.194 |
| Train+6000 | 0.254 | 0.159 | 0.255 | 0.254 | 0.255 | 0.095 | 0.547 | 0.198 |

The Result are around the same range for entire dataset However accuracy is not that efficient and hence not a good model.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **MultiLayer Perceptron** | TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area |
| Train | 0.341 | 0.163 | 0.31 | 0.341 | 0.308 | 0.174 | 0.664 | 0.308 |
| Test | 0.315 | 0.167 | 0.285 | 0.315 | 0.282 | 0.145 | 0.632 | 0.268 |
| Train+3000 | 0.341 | 0.163 | 0.31 | 0.341 | 0.308 | 0.174 | 0.664 | 0.308 |
| Train+6000 | 0.342 | 0.166 | 0.31 | 0.342 | 0.304 | 0.172 | 0.665 | 0.309 |

The neural network results are all in the same range, however the accuracy percentage is 30% which is not efficient.

**Comparative analysis of Neural Networks and Deep Neural Networks**

|  |  |
| --- | --- |
| Neural Network | Deep neural network (DNN) |
| Neural networks, a beautiful biologically-inspired programming paradigm which enables a computer to learn from observational data | A deep neural network (DNN) is an Artificial Neural Network (ANN) with multiple hidden layers between the input and output layers. Similar to shallow ANNs, DNNs can model complex non-linear relationships |
| provide the best solutions to many problems in image recognition, speech recognition, and natural language processing | provide the best solutions to many problems in image recognition, speech recognition, and natural language processing |
| Neural networks can be recurrent or feedforward; feedforward ones do not have any loops in their graph and can be organized in layers | If there are "many" layers, then we say that the network is deep. |
|  |  |
| Algortithms Available in Weka:   * multiLayerPerceptron * Java neural network package from unofficial weka * Learning Vector Quantizer   Packages/Algorithms Available in R:   * neuralnet | Packages/Algortithms Available in Weka:   * draft wrapper for deeplearning4j in WEKA’s SVN repository   Packages/Algorithms Available in R:   * MXNetR - Feed-forward neural network, convolutional neural network (CNN) * darch-Restricted Boltzmann machine, deep belief network * deepnet-Feed-forward neural network, restricted Boltzmann machine, deep belief network, stacked autoencoders * H2O-Feed-forward neural network, deep autoencoders * deepr-Simplify some functions from H2O and deepnet packages |

Ref: <http://neuralnetworksanddeeplearning.com/>

<https://en.wikipedia.org/wiki/Deep_learning#Deep_neural_networks>