Neural Network Ensembles

TRAINING DIFFERENT NEURAL NETWORK MODELS USING VARIOUS TOOLS TO SOLVE TWO DIFFERENT PROBLEMS

SHOBHIT JAIPURKAR – (A0163331R)

VIGNESH SRINIVASAN – (A0163246H)

M.TECH, KNOWLEDGE ENGINEERING | INSTITUTE OF SYSTEM SCIENCES, NUS

Introduction:

The primary aim of this assignment is to train different neural network models to achieve two tasks. Namely: Prediction of Wine Quality and the Diagnosis of Diabetes. The tools we used to implement the aforementioned were, RStudio and Weka. The packages used in RStudio were ***RCPP***, ***RSNNS***, ***neuralnet***, ***pnn*** and ***nnet***. We used the Multilayer Perceptron model with RBF in Weka.

Data Description:

The two main data sets we used were: WineQuality.csv and Diabetes.csv.

- ### Diabetes.csv

This the PIMA Indian Diabetes Database from the National Institute of Diabetes and Digestive and Kidney Diseases. It has 8 different attributes plus a Class Variable that defines whether or not the patient has diabetes. Those attributes are: Number of times Pregnant, Plasma Glucose Concentration, Diastolic Blood Pressure, Triceps Skin Fold Thickness, 2 Hour Serum Insulin, Body Mass Index, Diabetes Pedigree Function and Age.

While going through the data, we found a bunch of outliers in several columns. It is known that values such as BMI and Plasma Glucose Concentration cannot be 0. However, there are various values in those columns that are null. To clean the data, we deleted those values and calculated the mean of the respective columns. Then, we imputed the values of the same into these columns to go forward with our modelling assignments.

- ### Winequality.csv

The winequality data describes 11 different attributes corresponding to the quality of the wine. These attributes are Fixed Acidity, Volatile Acidity, Citric Acid, Residual Sugar, Chloride, Free and Sulfur Dioxide, Density, pH, Sulphates and Alcohol content. The Target Variable has a range from 3 to 9.

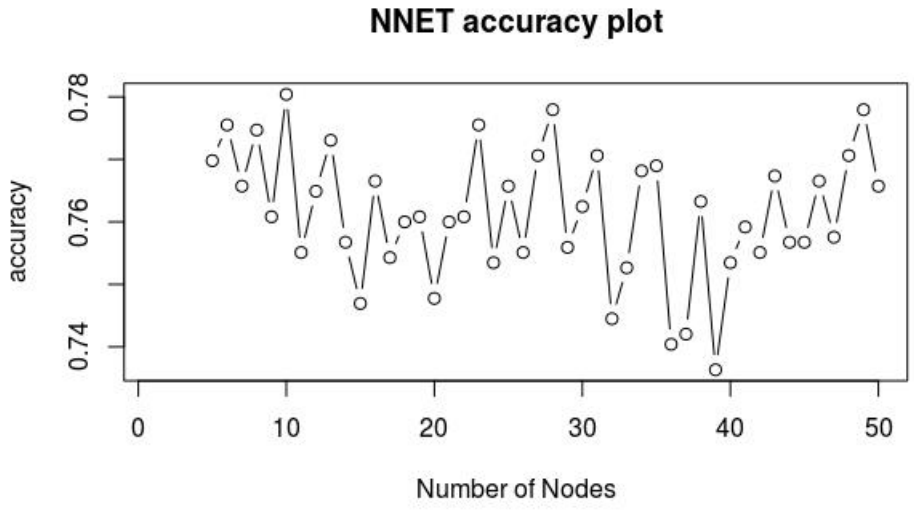
To normalize the given data, we have converted the Target Variable into a binary where the values from 3 to 6 correspond to **Bad Quality** and the values 7 to 9 correspond to **Good Quality.**

- The Data is then split into training and test partitions in the ratio of 3:4, or 75% to 25%.
- We modelled 4 different types of neural network using the stacking ensemble method, where we combined all the models and predicted target variables using Logistic regression

Results:

Wine Dataset Outputs

Single Layer Feed Forward (nnet) -
To identify the ideal number of nodes required in the hidden layer for the maximum accuracy, we shuffled through a various number of nodes and plot its corresponding accuracy to figure out the ideal node count. Below is a chart between Number of Nodes VS Accuracy



The accuracy is 78% when the number of units in hidden layer is 10

(**0 - Bad Quality, 1- Good Quality**)

true	0	1
0	283	132
1	137	673

PNN -

With the Probabilistic Neural Network approach, it was found that the accuracy of model decreases with an increase in sigma value while predicting the smoothness.

Sigma (0.1)

categories	0	1
0	314	242
1	101	568

Accuracy - 72%

Sigma (0.2)

categories	0	1
0	318	298
1	97	512

Accuracy- 67.7%

Sigma (0.3)

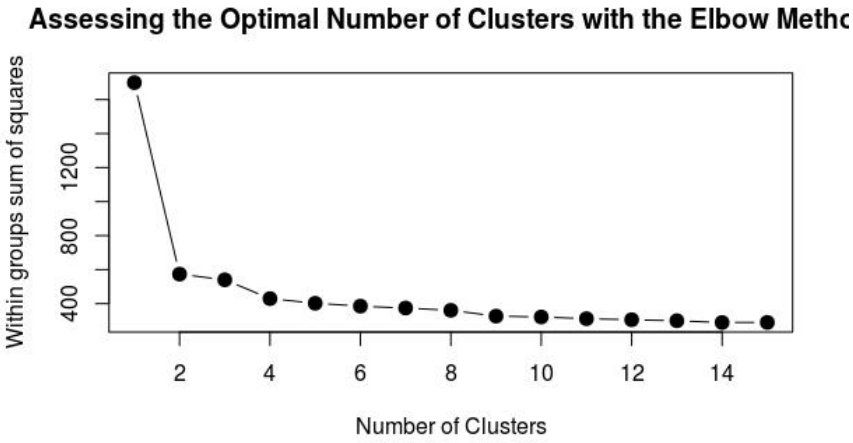
categories	0	1
0	321	305
1	94	505

Accuracy- 67.4%

RBF Neural Network -

As opposed to the computationally heavy method of selecting the number of nodes in the hidden layer, we can also employ K-mean clustering for deciding the number of nodes. Once we select the ideal number of nodes using the elbow method, the RBF model is applied with size equal to optimum number of clusters. Here, we observe that the ideal number of clusters is somewhere close to 4.

K-means Clustering Observations –

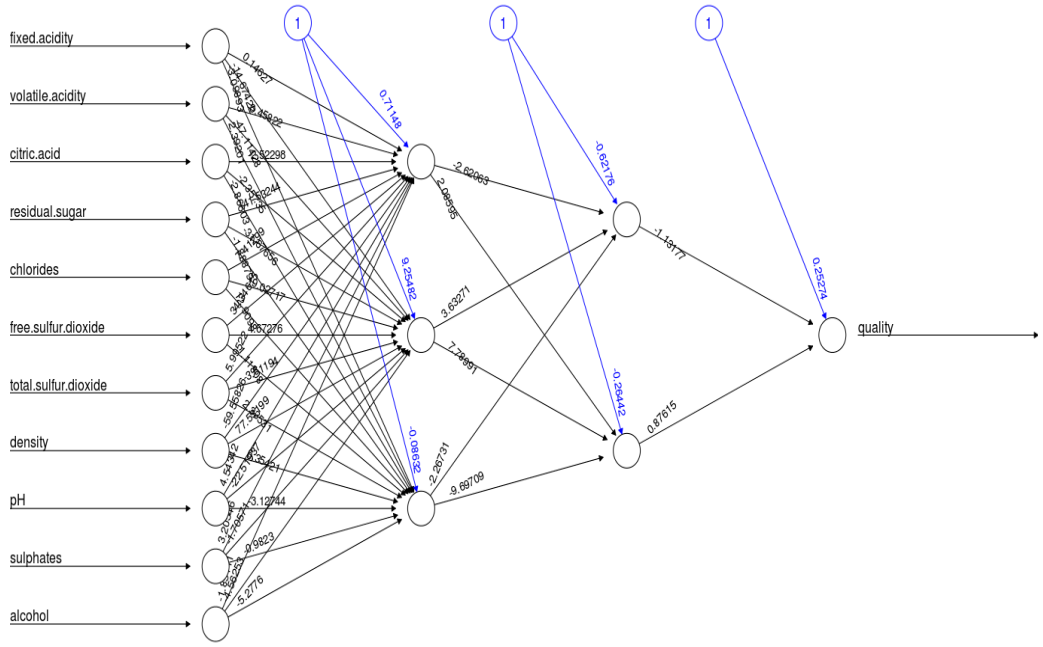


The accuracy with size = 4 is around **66.12%**

MLFF + BP:
The accuracy increases with number of hidden units in hidden layer. We were able to achieve 78 % accuracy with 3 hidden units and 2 hidden layers

Confusion Matrix (target vs output) -

	0	1
0	262	153
1	130	680



The summary of the outputs using Weka is detailed below:

Using Multilayer Perceptron

```
=== Summary ===

Correctly Classified Instances         920               75.1634 %
Incorrectly Classified Instances       304               24.8366 %
Kappa statistic                          0.4074
Mean absolute error                      0.3086
Root mean squared error                  0.4191
Relative absolute error                 68.7098 %
Root relative squared error             87.7606 %
Total Number of Instances             1224

=== Confusion Matrix ===

   a   b   <-- classified as
 717  78 |   a = good
 226 203 |   b = bad
```

Using RBF

```
=== Summary ===

Correctly Classified Instances         870               71.0784 %
Incorrectly Classified Instances       354               28.9216 %
Kappa statistic                          0.3257
Mean absolute error                      0.3552
Root mean squared error                  0.4271
Relative absolute error                 79.0992 %
Root relative squared error             89.4327 %
Total Number of Instances             1224
```

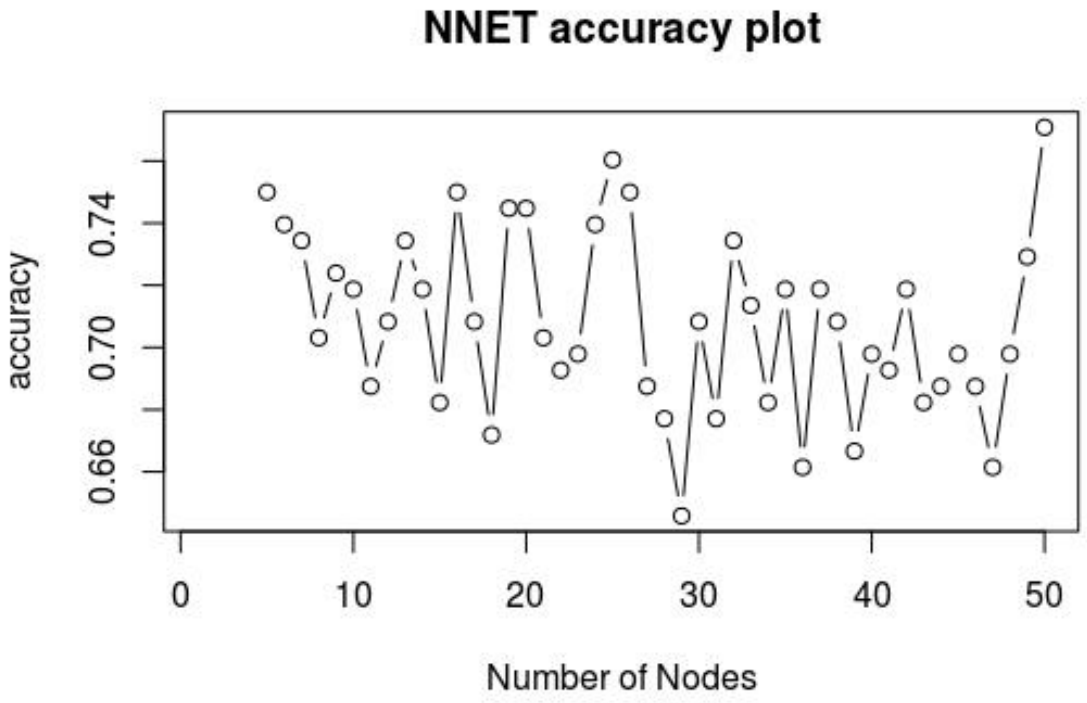
```
=== Confusion Matrix ===

   a   b   <-- classified as
 672 123 |   a = good
 231 198 |   b = bad
```

Diabetes Dataset Outputs:

Single Layer Feed Forward (nnet) –

To identify the ideal number of nodes required in the hidden layer for the maximum accuracy, we shuffled through a various number of nodes and plot its corresponding accuracy to figure out the ideal node count. Below is a chart between Number of Nodes VS Accuracy



The accuracy is **77%** when the number of units in hidden layer is **50**

```
true   0   1
   0 102  26
   1  18  46
```

PNN -

With the Probabilistic Neural Network approach, we found the accuracy to be highest for the Sigma value of 0.2

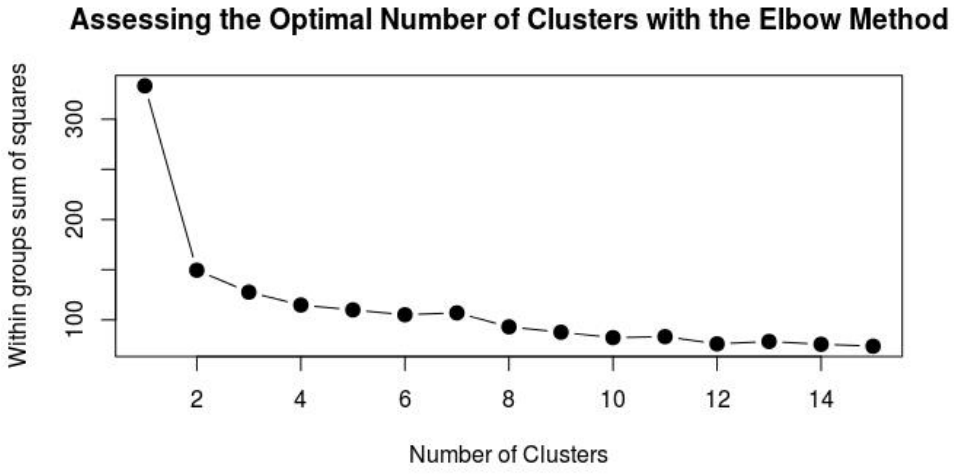
Sigma (0.2) -

```
categories   0   1
         0 100  18
         1  28  46
```

Accuracy: <u>76.04%</u>

RBF Neural Network -

After applying the K-means clustering to figure out the ideal number of hidden layer nodes, we apply the RBF model with the size equal to the optimal number of nodes. Using the Elbow method to find the best case value, we find that the optimal value of clusters should be 3. The accuracy corresponding to 3 sets of hidden layer nodes gives us 78.3%.



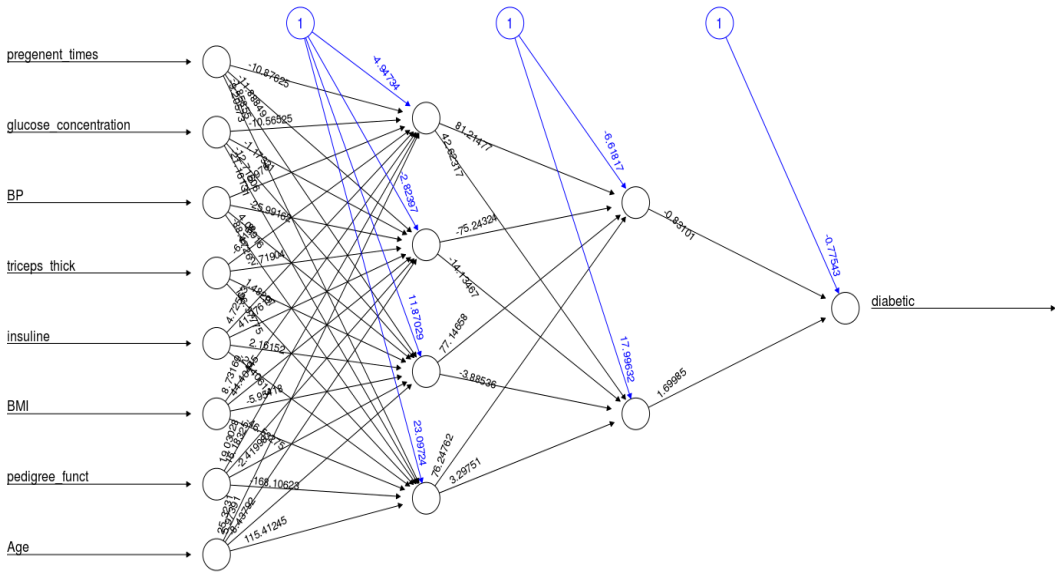
	FALSE	TRUE
0	113	15
1	27	37

MLFF + BP:

The accuracy increases with number of hidden units in hidden layer. We were able to achieve **77.6%** accuracy with 4 hidden units and 2 hidden layers

Confusion Matrix (Target vs Output) **-**

	0	1
0	111	17
1	26	38



The outputs for the Diabetes dataset using Weka are shown below:

Using RBF:

```
=== Summary ===

Correctly Classified Instances         149               77.6042 %
Incorrectly Classified Instances        43               22.3958 %
Kappa statistic                          0.4813
Mean absolute error                      0.3329
Root mean squared error                  0.3882
Relative absolute error                 74.0203 %
Root relative squared error             82.7927 %
Total Number of Instances              192

=== Confusion Matrix ===

   a   b   <-- classified as
  39  23 |   a = diabetes
  20 110 |   b = nodiabetes
```

Using Multi-Layer Perceptron

```
=== Summary ===

Correctly Classified Instances         152               79.1667 %
Incorrectly Classified Instances        40               20.8333 %
Kappa statistic                          0.4937
Mean absolute error                      0.2856
Root mean squared error                  0.4078
Relative absolute error                 63.5094 %
Root relative squared error             86.9573 %
Total Number of Instances              192
```

```
=== Confusion Matrix ===

   a   b   <-- classified as
  35  27 |   a = diabetes
  13 117 |   b = nodiabetes
```

Ensemble Approaches:

To further improve accuracy of the model using an ensemble Neural Network model, we employed the method of stacking as an ensembling technique. This was in attempts to increase the predictive accuracy of the model.

The combination of Multilayer perceptron, PNN and RBF prediction outcomes were then fed into a logistic regression model which gives the prediction output.

- **Wine Dataset:**
    - Using R Studio and R

	predicted	
true	FALSE	TRUE
0	293	126
1	200	606

Accuracy - **73.3%**

    - Using Weka

```
=== Summary ===

Correctly Classified Instances         921               75.2451 %
Incorrectly Classified Instances       303               24.7549 %
Kappa statistic                          0.429
Mean absolute error                      0.3315
Root mean squared error                  0.4157
Relative absolute error                 73.8081 %
Root relative squared error             87.0378 %
Total Number of Instances             1224

=== Confusion Matrix ===

   a   b   <-- classified as
 688 107 |   a = good
 196 233 |   b = bad
```

- **Diabetes Dataset:**
    - Using R Studio and R

	predicted	
true	FALSE	TRUE
0	117	11
1	37	27

Accuracy - **75%**

- Using Weka

```
=== Summary ===
Correctly Classified Instances         154               80.2083 %
Incorrectly Classified Instances        38               19.7917 %
Kappa statistic                          0.5396
Mean absolute error                      0.3139
Root mean squared error                  0.3872
Relative absolute error                 69.8023 %
Root relative squared error             82.567  %
Total Number of Instances              192

=== Confusion Matrix ===
   a   b   <-- classified as
  41  21 |   a = diabetes
  17 113 |   b = nodiabetes
```

Conclusion:

- We found that both, MLFF+BP and MLP perform better than any other model over the data, with an accuracy of 77%
- It was observed that stacking increases the predictive power, and thus provides us with a solution better than all the aforementioned models.