ANALYSIS AND CLASSIFICATION OF WINE REGIONS

Miguel Ángel Juárez Garzón

Data Load

```
datawine<-read.csv("wine.csv",header=FALSE)
colnames(datawine)<-c("Cultivars","Alcohol","Malic_acid","Ash","Alcalinity_of_ash",
                      "Magnesium","Total_phenols","Flavanoids","Nonflavanoid_phenol
s",
                      "Proanthocyanins","Color_intensity","Hue","OD280_OD315","Prolin
e")
head(datawine)
```

	Cultivars <int>	Alcohol <dbl>	Malic_acid <dbl>	A... <dbl>	Alcalinity_of_ash <dbl>	Magnesi... <int>	Total_phenols <dbl>	Flava
1	1	14.23	1.71	2.43	15.6	127	2.80	
2	1	13.20	1.78	2.14	11.2	100	2.65	
3	1	13.16	2.36	2.67	18.6	101	2.80	
4	1	14.37	1.95	2.50	16.8	113	3.85	
5	1	13.24	2.59	2.87	21.0	118	2.80	
6	1	14.20	1.76	2.45	15.2	112	3.27	

6 rows | 1-9 of 15 columns

```
summary(datawine)
```

```
##    Cultivars        Alcohol        Malic_acid         Ash       
##  Min.   :1.000   Min.   :11.03   Min.   :0.740   Min.   :1.360  
##  1st Qu.:1.000   1st Qu.:12.36   1st Qu.:1.603   1st Qu.:2.210  
##  Median :2.000   Median :13.05   Median :1.865   Median :2.360  
##  Mean   :1.938   Mean   :13.00   Mean   :2.336   Mean   :2.367  
##  3rd Qu.:3.000   3rd Qu.:13.68   3rd Qu.:3.083   3rd Qu.:2.558  
##  Max.   :3.000   Max.   :14.83   Max.   :5.800   Max.   :3.230  
##  Alcalinity_of_ash   Magnesium      Total_phenols     Flavanoids   
##  Min.   :10.60     Min.   : 70.00   Min.   :0.980   Min.   :0.340  
##  1st Qu.:17.20     1st Qu.: 88.00   1st Qu.:1.742   1st Qu.:1.205  
##  Median :19.50     Median : 98.00   Median :2.355   Median :2.135  
##  Mean   :19.49     Mean   : 99.74   Mean   :2.295   Mean   :2.029  
##  3rd Qu.:21.50     3rd Qu.:107.00   3rd Qu.:2.800   3rd Qu.:2.875  
##  Max.   :30.00     Max.   :162.00   Max.   :3.880   Max.   :5.080  
##  Nonflavanoid_phenols Proanthocyanins Color_intensity       Hue        
##  Min.   :0.1300       Min.   :0.410   Min.   : 1.280   Min.   :0.4800  
##  1st Qu.:0.2700       1st Qu.:1.250   1st Qu.: 3.220   1st Qu.:0.7825  
##  Median :0.3400       Median :1.555   Median : 4.690   Median :0.9650  
##  Mean   :0.3619       Mean   :1.591   Mean   : 5.058   Mean   :0.9574  
##  3rd Qu.:0.4375       3rd Qu.:1.950   3rd Qu.: 6.200   3rd Qu.:1.1200  
##  Max.   :0.6600       Max.   :3.580   Max.   :13.000   Max.   :1.7100  
##   OD280_OD315       Proline      
##  Min.   :1.270   Min.   : 278.0  
##  1st Qu.:1.938   1st Qu.: 500.5  
##  Median :2.780   Median : 673.5  
##  Mean   :2.612   Mean   : 746.9  
##  3rd Qu.:3.170   3rd Qu.: 985.0  
##  Max.   :4.000   Max.   :1680.0
```

Libraries

```
library(neuralnet)
library(nnet)
library(NeuralNetTools)
```

Exercise 1: Set generation

```
ndf <- length(datawine$Cultivars)
ndf
```

```
## [1] 178
```

```
#Random sample
wine_splice <- sample(ndf,0.7 * ndf,replace = FALSE)
# Creating training and test dataset
dftrain <- datawine[wine_splice,]
dftest <- datawine[-wine_splice,]
dftrain
```

	Cultivars	Alcohol	Malic_acid	A…	Alcalinity_of_ash	Magnesi…	Total_phenols	Fla
	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<dbl>	
68	2	12.37	1.17	1.92	19.6	78	2.11	
167	3	13.45	3.70	2.60	23.0	111	1.70	
129	2	12.37	1.63	2.30	24.5	88	2.22	
162	3	13.69	3.26	2.54	20.0	107	1.83	
43	1	13.88	1.89	2.59	15.0	101	3.25	
14	1	14.75	1.73	2.39	11.4	91	3.10	
51	1	13.05	1.73	2.04	12.4	92	2.72	
85	2	11.84	0.89	2.58	18.0	94	2.20	
21	1	14.06	1.63	2.28	16.0	126	3.00	
106	2	12.42	2.55	2.27	22.0	90	1.68	

1-10 of 124 rows | 1-9 of 15 columns Previous **1** 2 3 4 5 6 ... 13 Next

```
dftest
```

	Cultivars	Alcohol	Malic_acid	A…	Alcalinity_of_ash	Magnesi…	Total_phenols	Fla
	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<dbl>	
2	1	13.20	1.78	2.14	11.2	100	2.65	
3	1	13.16	2.36	2.67	18.6	101	2.80	
4	1	14.37	1.95	2.50	16.8	113	3.85	
5	1	13.24	2.59	2.87	21.0	118	2.80	
6	1	14.20	1.76	2.45	15.2	112	3.27	
8	1	14.06	2.15	2.61	17.6	121	2.60	
9	1	14.83	1.64	2.17	14.0	97	2.80	
11	1	14.10	2.16	2.30	18.0	105	2.95	
12	1	14.12	1.48	2.32	16.8	95	2.20	
16	1	13.63	1.81	2.70	17.2	112	2.85	

1-10 of 54 rows | 1-9 of 15 columns Previous **1** 2 3 4 5 6 Next

```
ntrain <- length(dftrain$Cultivars)
ntest <- length(dftest$Cultivars)

cat("Train set size: ", ntrain, "\n")
```

```
## Train set size:  124
```

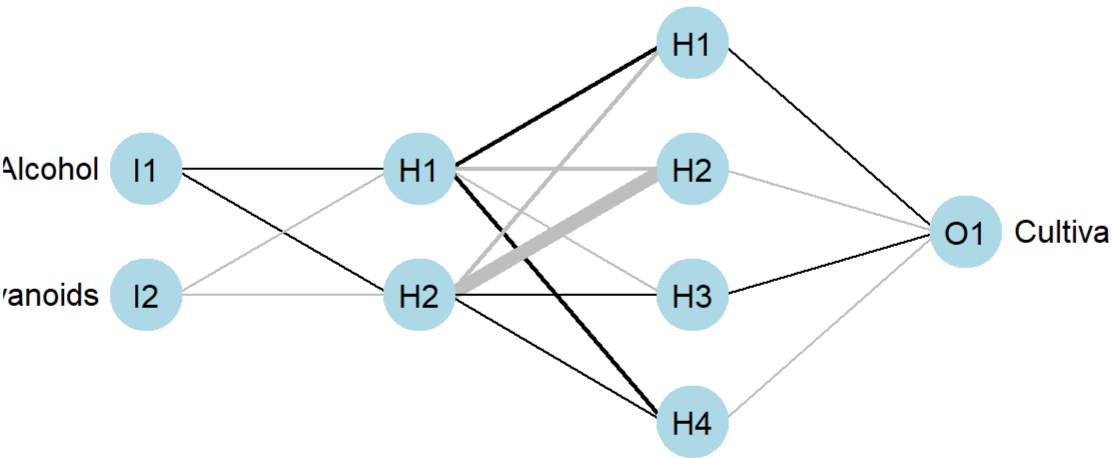
```
cat("Test set size: ", ntest, "\n")
```

```
## Test set size:  54
```

Exercise 2a: Construction and training

```
#Creation of neural network using neuralnet
nn <- neuralnet(Cultivars ~ Alcohol + Flavanoids, dftrain, hidden = c(2,4), linear.ou
tput = TRUE)

#Network drawing
plotnet(nn, bias = FALSE)
```




```
prediction(nn)
```

```
## Data Error:  0;
```

```
## $rep1
##     Alcohol Flavanoids Cultivars
## 1     13.88       0.34 2.9731523
## 2     12.25       0.47 2.9731540
## 3     13.73       0.47 2.9731523
## 4     13.49       0.48 2.9731523
## 5     12.93       0.50 2.9731523
## 6     13.36       0.50 2.9731523
## 7     12.77       0.51 2.9731523
## 8     13.16       0.55 2.9731523
## 9     13.69       0.56 2.9731523
## 10    12.37       0.57 2.9731530
## 11    12.45       0.58 2.9731524
## 12    12.51       0.58 2.9731523
## 13    12.58       0.58 2.9731523
## 14    12.53       0.60 2.9731523
## 15    12.84       0.60 2.9731523
## 16    13.71       0.61 2.9731523
## 17    13.17       0.63 2.9731523
## 18    12.87       0.65 2.9731523
## 19    12.82       0.66 2.9731523
## 20    13.27       0.69 2.9731523
## 21    12.96       0.70 2.9731523
## 22    14.16       0.70 2.9731523
## 23    13.40       0.75 2.9731523
## 24    13.32       0.76 2.9731523
## 25    12.25       0.78 2.9732315
## 26    13.62       0.80 2.9731523
## 27    13.84       0.83 2.9731523
## 28    13.58       0.84 2.9731523
## 29    12.36       0.92 2.9704081
## 30    13.45       0.92 2.9731521
## 31    13.40       0.96 2.9731515
## 32    11.81       0.99 1.9574940
## 33    12.33       1.09 2.4435521
## 34    12.81       1.09 2.9659342
## 35    12.70       1.20 2.5263369
## 36    12.77       1.25 2.2967564
## 37    12.86       1.25 2.6463853
## 38    12.21       1.28 1.9914519
## 39    13.11       1.28 3.0167533
## 40    12.60       1.36 2.1734774
## 41    12.79       1.36 2.7912850
## 42    12.69       1.46 2.0096873
## 43    11.66       1.57 2.0062112
## 44    13.50       1.57 3.0905737
## 45    12.08       1.58 2.0041035
## 46    12.08       1.59 2.0029866
## 47    13.05       1.59 1.9898658
## 48    11.82       1.60 2.0063012
## 49    11.82       1.64 2.0063059
## 50    12.00       1.64 2.0056286
## 51    11.64       1.69 2.0063035
## 52    12.16       1.69 2.0006636
## 53    12.04       1.75 2.0057248
```

```
## 54    12.72       1.76 1.9983384
## 55    13.67       1.79 1.9836010
## 56    12.42       1.84 1.9219615
## 57    13.49       1.84 1.9588127
## 58    12.33       1.85 1.9847757
## 59    12.51       1.92 1.8384437
## 60    12.37       2.00 1.9918476
## 61    11.41       2.01 2.0062796
## 62    12.22       2.04 2.0049781
## 63    12.34       2.11 2.0013395
## 64    12.42       2.13 1.9923783
## 65    11.96       2.14 2.0062870
## 66    11.03       2.17 2.0050545
## 67    13.30       2.19 0.9946121
## 68    11.84       2.21 2.0063046
## 69    12.29       2.25 2.0051180
## 70    11.62       2.26 2.0063051
## 71    12.00       2.26 2.0062857
## 72    12.52       2.27 1.9789207
## 73    12.08       2.29 2.0062516
## 74    12.85       2.37 1.3230556
## 75    12.93       2.41 1.1489499
## 76    12.37       2.45 2.0048820
## 77    12.29       2.50 2.0059239
## 78    14.39       2.52 1.0001396
## 79    13.51       2.53 0.9944874
## 80    11.46       2.58 2.0062879
## 81    13.50       2.61 0.9945269
## 82    13.24       2.63 0.9997528
## 83    13.07       2.64 1.0616965
## 84    12.07       2.65 2.0062945
## 85    14.21       2.65 0.9944387
## 86    13.05       2.68 1.0995748
## 87    13.28       2.68 0.9981668
## 88    13.41       2.68 0.9949553
## 89    13.68       2.69 0.9944443
## 90    13.76       2.74 0.9944385
## 91    13.75       2.76 0.9944393
## 92    13.56       2.78 0.9945218
## 93    11.45       2.79 2.0062739
## 94    13.77       2.79 0.9944385
## 95    13.71       2.88 0.9944487
## 96    12.99       2.89 1.4178649
## 97    13.74       2.90 0.9944443
## 98    11.61       2.92 2.0062989
## 99    14.10       2.92 0.9944344
## 100   13.87       2.97 0.9944360
## 101   12.29       2.99 2.0062497
## 102   13.05       3.00 1.3228809
## 103   13.64       3.03 0.9945182
## 104   14.22       3.04 0.9944339
## 105   14.23       3.06 0.9944338
## 106   12.37       3.10 2.0061989
## 107   14.30       3.14 0.9944336
## 108   12.43       3.15 2.0061091
## 109   13.86       3.15 0.9944388
```

```
## 110   14.06       3.17 0.9944338
## 111   13.11       3.18 1.2965347
## 112   13.29       3.23 1.0309951
## 113   13.73       3.25 0.9944911
## 114   13.05       3.27 1.5959408
## 115   13.56       3.29 0.9953149
## 116   13.90       3.39 0.9944414
## 117   13.94       3.54 0.9944416
## 118   13.88       3.56 0.9944558
## 119   14.38       3.64 0.9944331
## 120   13.72       3.67 0.9948282
## 121   14.75       3.69 0.9944329
## 122   13.82       3.74 0.9945511
## 123   12.37       3.75 2.0062946
## 124   11.56       5.08 1.9996572
##
## $data
##     Alcohol Flavanoids Cultivars
## 1     13.88       0.34         3
## 2     12.25       0.47         3
## 3     13.73       0.47         3
## 4     13.49       0.48         3
## 5     12.93       0.50         3
## 6     13.36       0.50         3
## 7     12.77       0.51         3
## 8     13.16       0.55         3
## 9     13.69       0.56         3
## 10    12.37       0.57         2
## 11    12.45       0.58         3
## 12    12.51       0.58         3
## 13    12.58       0.58         3
## 14    12.53       0.60         3
## 15    12.84       0.60         3
## 16    13.71       0.61         3
## 17    13.17       0.63         3
## 18    12.87       0.65         3
## 19    12.82       0.66         3
## 20    13.27       0.69         3
## 21    12.96       0.70         3
## 22    14.16       0.70         3
## 23    13.40       0.75         3
## 24    13.32       0.76         3
## 25    12.25       0.78         3
## 26    13.62       0.80         3
## 27    13.84       0.83         3
## 28    13.58       0.84         3
## 29    12.36       0.92         3
## 30    13.45       0.92         3
## 31    13.40       0.96         3
## 32    11.81       0.99         2
## 33    12.33       1.09         2
## 34    12.81       1.09         3
## 35    12.70       1.20         3
## 36    12.77       1.25         2
## 37    12.86       1.25         3
## 38    12.21       1.28         2
```

```
## 39    13.11       1.28          3
## 40    12.60       1.36          2
## 41    12.79       1.36          3
## 42    12.69       1.46          2
## 43    11.66       1.57          2
## 44    13.50       1.57          3
## 45    12.08       1.58          2
## 46    12.08       1.59          2
## 47    13.05       1.59          2
## 48    11.82       1.60          2
## 49    11.82       1.64          2
## 50    12.00       1.64          2
## 51    11.64       1.69          2
## 52    12.16       1.69          2
## 53    12.04       1.75          2
## 54    12.72       1.76          2
## 55    13.67       1.79          2
## 56    12.42       1.84          2
## 57    13.49       1.84          2
## 58    12.33       1.85          2
## 59    12.51       1.92          2
## 60    12.37       2.00          2
## 61    11.41       2.01          2
## 62    12.22       2.04          2
## 63    12.34       2.11          2
## 64    12.42       2.13          2
## 65    11.96       2.14          2
## 66    11.03       2.17          2
## 67    13.30       2.19          1
## 68    11.84       2.21          2
## 69    12.29       2.25          2
## 70    11.62       2.26          2
## 71    12.00       2.26          2
## 72    12.52       2.27          2
## 73    12.08       2.29          2
## 74    12.85       2.37          1
## 75    12.93       2.41          1
## 76    12.37       2.45          2
## 77    12.29       2.50          2
## 78    14.39       2.52          1
## 79    13.51       2.53          1
## 80    11.46       2.58          2
## 81    13.50       2.61          1
## 82    13.24       2.63          1
## 83    13.07       2.64          1
## 84    12.07       2.65          2
## 85    14.21       2.65          1
## 86    13.05       2.68          1
## 87    13.28       2.68          1
## 88    13.41       2.68          1
## 89    13.68       2.69          1
## 90    13.76       2.74          1
## 91    13.75       2.76          1
## 92    13.56       2.78          1
## 93    11.45       2.79          2
## 94    13.77       2.79          1
```

```
## 95    13.71       2.88         1
## 96    12.99       2.89         2
## 97    13.74       2.90         1
## 98    11.61       2.92         2
## 99    14.10       2.92         1
## 100   13.87       2.97         1
## 101   12.29       2.99         2
## 102   13.05       3.00         1
## 103   13.64       3.03         1
## 104   14.22       3.04         1
## 105   14.23       3.06         1
## 106   12.37       3.10         2
## 107   14.30       3.14         1
## 108   12.43       3.15         2
## 109   13.86       3.15         1
## 110   14.06       3.17         1
## 111   13.11       3.18         2
## 112   13.29       3.23         1
## 113   13.73       3.25         1
## 114   13.05       3.27         1
## 115   13.56       3.29         1
## 116   13.90       3.39         1
## 117   13.94       3.54         1
## 118   13.88       3.56         1
## 119   14.38       3.64         1
## 120   13.72       3.67         1
## 121   14.75       3.69         1
## 122   13.82       3.74         1
## 123   12.37       3.75         2
## 124   11.56       5.08         2
```

Exercise 2b: Validation phase

```
# PPredict validation set and round predictions
pred <- predict(nn, dftest)
pred <- round(pred)

#Only values 1,2 or 3
for (i in 1:length(pred)){
  if (pred[i] == 4){
    pred[i] = 3
  }
}

#Contingency table
tb <- table(dftest$Cultivars, pred)
tb
```

```
##    pred
##      1  2  3
##   1 20  0  0
##   2  4 14  4
##   3  0  0 12
```

```
#Error rate
error_rate <- sum(sum(tb-diag(diag(tb))))/sum(colSums(tb))
error_rate
```

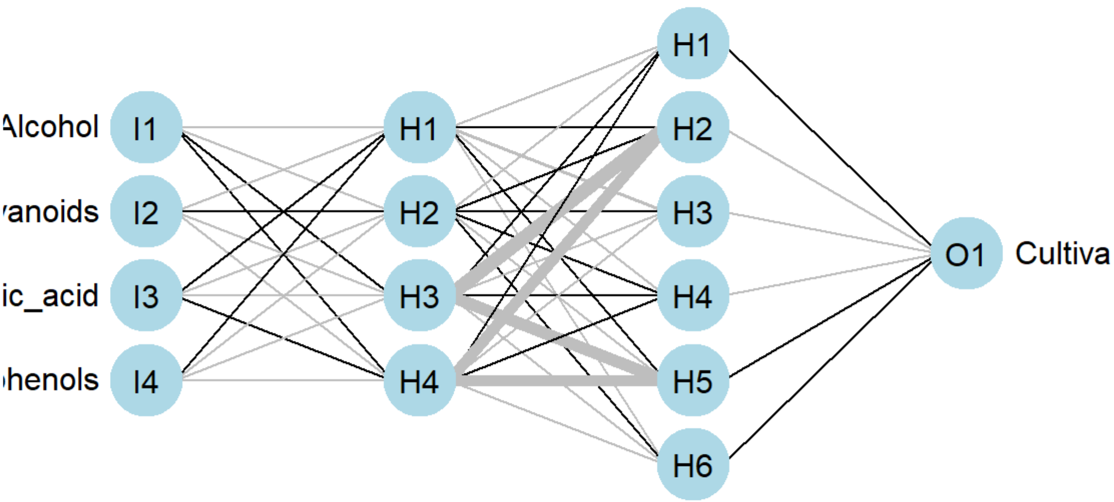
```
## [1] 0.1481481
```

Neural network acts as a classifier, hoever the error rate is a bit high. To reduce the error rate we can use more attributes or change the network architecture

Exercise 3: Effect of attributes over classification

```
nn_mod <- neuralnet(Cultivars ~ Alcohol + Flavanoids + Malic_acid + Total_phenols, df
train, hidden = c(4,6), linear.output = TRUE)
```

```
plotnet(nn_mod, bias = FALSE)
```



```
pred <- predict(nn_mod, dftest)
pred <- round(pred)

for (i in 1:length(pred)){
  if (pred[i] >= 4){
    pred[i] = 3
  } else if (pred[i] < 1) {
    pred[i] = 1
  }
}

tb <- table(dftest$Cultivars, pred)
tb
```

```
##    pred
##      1  2  3
##   1 19  1  0
##   2  2 18  2
##   3  0  1 11
```

```
#Error rate
error_rate <- sum(sum(tb-diag(diag(tb))))/sum(colSums(tb))
error_rate
```

```
## [1] 0.1111111
```

The error rate has decreased when increasing the number of attributes included as input of the network.

Exercise 4

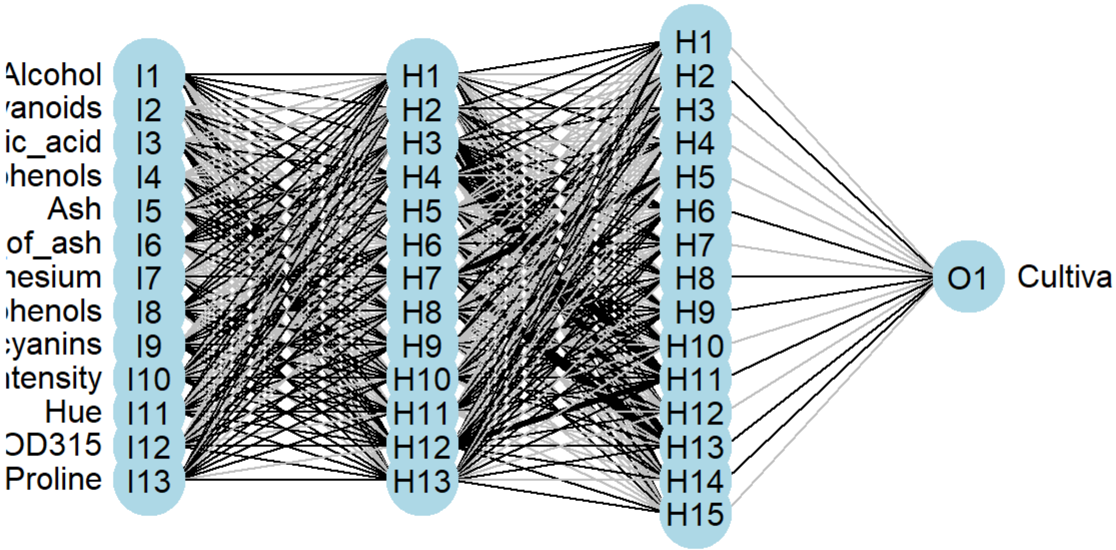
In this exercise, we are going to conduct a study on the effect of architecture on the classification capacity of the network. To do this, we will use the thirteen available attributes and compare different topologies by varying both the number of hidden layers and the number of neurons in those layers.

Basic Configuration

In this initial test, we will create a neural network using the same model we have been following so far. For this purpose, we will use two hidden layers, with the same number of neurons in the first layer as there are attributes, and two additional neurons in the second layer compared to the number of attributes. In this case, it would result in two hidden layers with 13 and 15 neurons, respectively.

```
nn_fin <- neuralnet(Cultivars ~ Alcohol + Flavanoids + Malic_acid + Total_phenols + A
sh + Alcalinity_of_ash + Magnesium + Nonflavanoid_phenols + Proanthocyanins + Color_i
ntensity + Hue + OD280_OD315 + Proline, dftrain, hidden = c(13,15), linear.output = T
RUE)
```

```
plotnet(nn_fin, bias = FALSE)
```




```
pred <- predict(nn_fin , dftest)
pred <- round(pred)


for (i in 1:length(pred)){
  if (pred[i] >= 4){
    pred[i] = 3
  } else if (pred[i] <= 0){
    pred[i] = 1
  }
}

tb <- table(dftest$Cultivars, pred)
tb
```

```
##    pred
##      1  2  3
##   1 19  1  0
##   2  3 17  2
##   3  0  0 12
```

```
#Error rate
error_rate <- sum(sum(tb-diag(diag(tb))))/sum(colSums(tb))
error_rate
```

```
## [1] 0.1111111
```

As we can see, this configuration does not improve the classification capacity of the neural network compared to the network that uses only four attributes and a significantly smaller number of neurons. Therefore, this structure is not providing us with any additional information.

Reducing neuron nuber of first layer

In this case, we will keep the number of hidden layers the same, but we will reduce the number of neurons in the first layer.

```
nn_fin <- neuralnet(Cultivars ~ Alcohol + Flavanoids + Malic_acid + Total_phenols + Ash + Alcalinity_of_ash + Magnesium + Nonflavanoid_phenols + Proanthocyanins + Color_intensity + Hue + OD280_OD315 + Proline, dftrain, hidden = c(7,15), linear.output = TRUE)
```

```
pred <- predict(nn_fin , dftest)
pred <- round(pred)


for (i in 1:length(pred)){
  if (pred[i] >= 4){
    pred[i] = 3
  } else if (pred[i] <= 0){
    pred[i] = 1
  }
}

tb <- table(dftest$Cultivars, pred)
tb
```

```
##    pred
##      2
##   1 20
##   2 22
##   3 12
```

```
#Error rate
error_rate <- (tb[1] + tb[3]) / (tb[1] + tb[2] + tb[3])
error_rate
```

```
## [1] 0.5925926
```

As we can observe, reducing the number of neurons in the first layer drastically decreases the classification capacity of the network, leading to a significant increase in the error rate. This indicates that the first layer plays a crucial role in the network's classification function, likely because it processes the input data initially.

Reduction of Neurons in the Second Layer

In this case, we will keep the number of hidden layers the same but reduce the number of neurons in the second layer.

```
nn_fin <- neuralnet(Cultivars ~ Alcohol + Flavanoids + Malic_acid + Total_phenols + A
sh + Alcalinity_of_ash + Magnesium + Nonflavanoid_phenols + Proanthocyanins + Color_i
ntensity + Hue + OD280_OD315 + Proline, dftrain, hidden = c(13,8), linear.output = TR
UE)
```

```
pred <- predict(nn_fin , dftest)
pred <- round(pred)


for (i in 1:length(pred)){
  if (pred[i] >= 4){
    pred[i] = 3
  } else if (pred[i] <= 0){
    pred[i] = 1
  }
}

tb <- table(dftest$Cultivars, pred)
tb
```

```
##    pred
##      1  2  3
##   1 19  1  0
##   2  5 16  1
##   3  0  0 12
```

```
#Error rate
error_rate <- sum(sum(tb-diag(diag(tb))))/sum(colSums(tb))
error_rate
```

```
## [1] 0.1296296
```

In this case, we can observe that there is not a significant change in the error rate. This suggests that the number of neurons in the second layer may not be as crucial for classification but rather serves as an adjustment function, contributing to slight improvements in the network's accuracy.

However, it is important to note that if we reduce the number of neurons in both layers, the neural network will lose its validity as a classifier.

Increase of neurons in teh first layer

In this case, we will keep the number of hidden layers the same but increase the number of neurons in the first layer.

```
nn_fin <- neuralnet(Cultivars ~ Alcohol + Flavanoids + Malic_acid + Total_phenols + A
sh + Alcalinity_of_ash + Magnesium + Nonflavanoid_phenols + Proanthocyanins + Color_i
ntensity + Hue + OD280_OD315 + Proline, dftrain, hidden = c(18,15), linear.output = T
RUE)
```

```
pred <- predict(nn_fin , dftest)
pred <- round(pred)


for (i in 1:length(pred)){
  if (pred[i] >= 4){
    pred[i] = 3
  } else if (pred[i] <= 0){
    pred[i] = 1
  }
}

tb <- table(dftest$Cultivars, pred)
tb
```

```
##    pred
##      1  2  3
##   1 20  0  0
##   2  2 18  2
##   3  0  0 12
```

```
#Error rate
error_rate <- sum(sum(tb-diag(diag(tb))))/sum(colSums(tb))
error_rate
```

```
## [1] 0.07407407
```

We can observe an improvement in the classification by the neural network. This confirms the assumption we made earlier, validating that the first hidden layer is of great importance for the network's function as it is responsible for processing the input data.

Increase of neurons in the second layer

In this case, we will keep the number of hidden layers the same but increase the number of neurons in the second layer.

```
nn_fin <- neuralnet(Cultivars ~ Alcohol + Flavanoids + Malic_acid + Total_phenols + A
sh + Alcalinity_of_ash + Magnesium + Nonflavanoid_phenols + Proanthocyanins + Color_i
ntensity + Hue + OD280_OD315 + Proline, dftrain, hidden = c(13,20), linear.output = T
RUE)
```

```
pred <- predict(nn_fin , dftest)
pred <- round(pred)


for (i in 1:length(pred)){
  if (pred[i] >= 4){
    pred[i] = 3
  } else if (pred[i] <= 0){
    pred[i] = 1
  }
}

tb <- table(dftest$Cultivars, pred)
tb
```

```
##    pred
##      1  2  3
##   1 19  1  0
##   2  2 18  2
##   3  0  0 12
```

```
#Error rate
error_rate <- sum(sum(tb-diag(diag(tb))))/sum(colSums(tb))
error_rate
```

```
## [1] 0.09259259
```

In this case, we also observe a decrease in the error rate, but it is less significant compared to the increase in the first layer. This confirms the assumption that the second layer serves as an adjustment and fine-tuning function for the results.

Increase of neurons in both layers

In this case, we will keep the number of hidden layers the same but increase the number of neurons in both layers.

```
nn_fin <- neuralnet(Cultivars ~ Alcohol + Flavanoids + Malic_acid + Total_phenols + A
sh + Alcalinity_of_ash + Magnesium + Nonflavanoid_phenols + Proanthocyanins + Color_i
ntensity + Hue + OD280_OD315 + Proline, dftrain, hidden = c(18,20), linear.output = T
RUE)
```

```
pred <- predict(nn_fin , dftest)
pred <- round(pred)


for (i in 1:length(pred)){
  if (pred[i] >= 4){
    pred[i] = 3
  } else if (pred[i] <= 0){
    pred[i] = 1
  }
}

tb <- table(dftest$Cultivars, pred)
tb
```

```
##    pred
#     1  2  3
#  1 19  1  0
#  2  0 21  1
#  3  0  0 12
```

```
#Error rate
error_rate <- sum(sum(tb-diag(diag(tb))))/sum(colSums(tb))
error_rate
```

```
## [1] 0.03703704
```

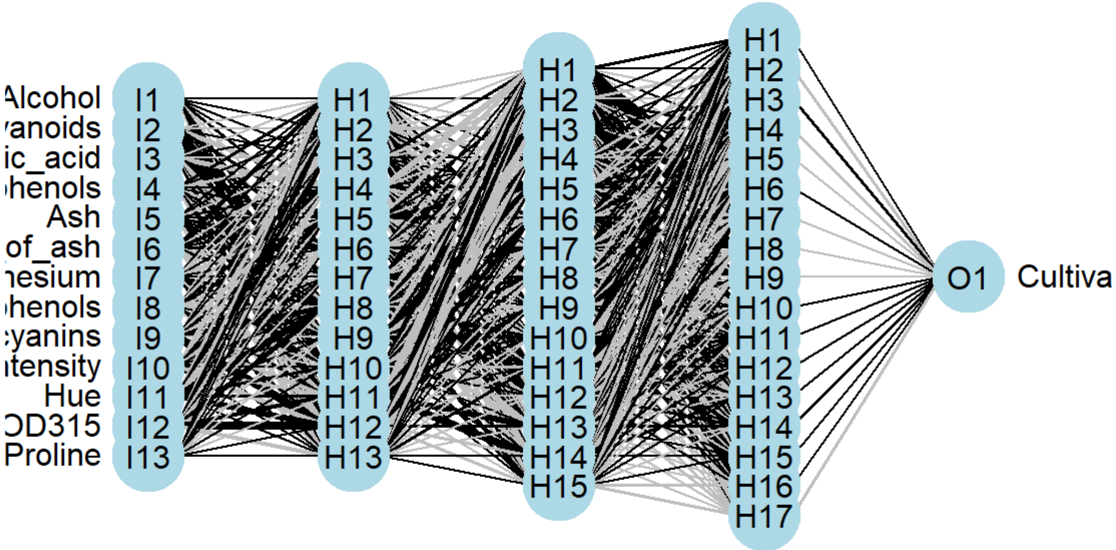
Increasing the number of neurons in both layers leads to a substantial improvement in the network. This is understandable since we are enhancing both the initial processing of the input data and the final adjustment and fine-tuning of the results. Therefore, the combination of these changes provides a significant enhancement in the classification.

Increase the number of hidden layers

In this case, we will increase the number of hidden layers while keeping the number of neurons in the previous layers the same.

```
nn_fin <- neuralnet(Cultivars ~ Alcohol + Flavanoids + Malic_acid + Total_phenols + A
sh + Alcalinity_of_ash + Magnesium + Nonflavanoid_phenols + Proanthocyanins + Color_i
ntensity + Hue + OD280_OD315 + Proline, dftrain, hidden = c(13,15,17), linear.output
= TRUE)
```

```
plotnet(nn_fin, bias = FALSE)
```




```
pred <- predict(nn_fin , dftest)
pred <- round(pred)


for (i in 1:length(pred)){
  if (pred[i] >= 4){
    pred[i] = 3
  } else if (pred[i] <= 0){
    pred[i] = 1
  }
}

tb <- table(dftest$Cultivars, pred)
tb
```

```
##    pred
##      1  2  3
##   1 20  1  0
##   2  0 21  1
##   3  0  0 12
```

```
#Error rate
error_rate <- sum(sum(tb-diag(diag(tb))))/sum(colSums(tb))
error_rate
```

```
## [1] 0.07407407
```

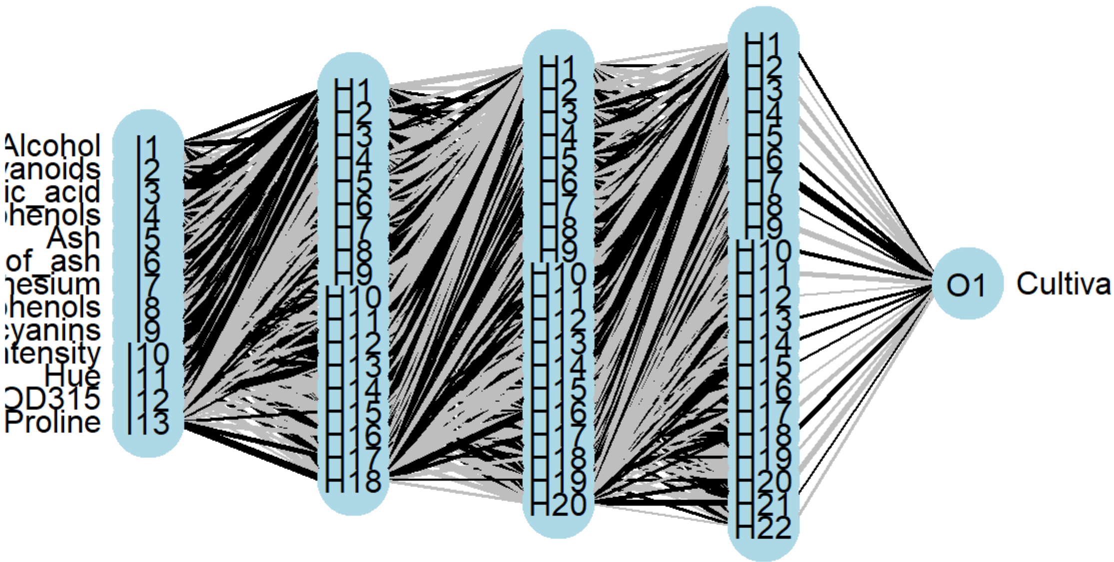
We can observe how increasing the number of hidden layers also results in an improvement in the classifier based on the neural network. This is possibly because it introduces another point of refinement in the predictions.

Increasing number of layers and neurons

In this case, we will increase the number of hidden layers and also increase the number of neurons per layer.

```
nn_fin <- neuralnet(Cultivars ~ Alcohol + Flavanoids + Malic_acid + Total_phenols + A
sh + Alcalinity_of_ash + Magnesium + Nonflavanoid_phenols + Proanthocyanins + Color_i
ntensity + Hue + OD280_OD315 + Proline, dftrain, hidden = c(18,20,22), linear.output
= TRUE, rep = 5)
```

```
plotnet(nn_fin, bias = FALSE)
```



```
pred <- predict(nn_fin , dftest)
pred <- round(pred)


for (i in 1:length(pred)){
  if (pred[i] >= 4){
    pred[i] = 3
  } else if (pred[i] <= 0){
    pred[i] = 1
  }
}

tb <- table(dftest$Cultivars, pred)
tb
```

```
##    pred
#     1  2  3
#  1 20  0  0
#  2  2 17  3
#  3  2  0 10
```

```
#Error rate
error_rate <- sum(sum(tb-diag(diag(tb))))/sum(colSums(tb))
error_rate
```

```
## [1] 0.1296296
```

In this case, we do not observe an improvement in the classification, which is somewhat contradictory considering the results obtained so far. It is possible that by introducing more layers and neurons, we are overfitting the network, leading to a loss in classification effectiveness. However, it is also possible that this result is due to the randomness involved in generating the neural network.

Conclusions

From the analysis conducted, we can draw several interesting conclusions that can guide us when performing other classifications using neural networks. These conclusions are as follows:

1. The first hidden layer is of great importance in the processing of input data and has the most significant effect on classification improvement. The larger the number of neurons in this layer, the better the initial data processing.

2. Subsequent layers are of lesser importance and are responsible for adjusting the results, generating more precise classification and providing the final touches to the predictions.

3. Increasing the number of layers leads to an improvement in classification as it introduces more levels where small adjustments to the output are made.

4. An excessive number of layers or neurons can lead to network collapse and a loss of effectiveness.

Additionally, there is a certain level of randomness involved in generating the neural network. Therefore, it is advisable to generate multiple repetitions of the network and select the best-performing one.