STATISTICAL LEARNING

Home Assignment 2

JUNE 5, 2022

KRISHNA BHARAT PASUMARTHY
h21pasbh@du.se

Introduction:

The Data_Cortex_Nuclear.xls data has the expression levels of 77 proteins, detectable in the nuclear fraction of cortex of mice. The experiment was performed on 72 mice belonging to 2 categories control and down syndrome. Apart form 77 protein columns there are 4 class variables and 1 key column MouseID. Note that MouseID column is a just used for identifying the mice uniquely, so that column was not considered in any part of the analysis. Loaded the xls file into R, created a data frame 'df' and analyzed the column information, summary, and structure of the dataset.

Pre-Processing:

Firstly, checked for NA values in the columns and found out that we have a total of 1396 NA values in all the 1080 rows together. I have approached to replace the NA values with the median values of that specific column, separately for each of the 8 class categories in class column. So, I have first split he dataframe based on the class, so all the rows are split into 8 classes grouped together. After replacing NAs with median, I have combined and shuffled the dataset (shuffled_df) and this is used further for solving questions 1 and 2.

Question 1.A

Method:

Using the shuffled_df dataframe, created 4 dataframes dataset1, dataset2, dataset3, dataset4 having 77 protein columns in common and only 1 class column out of 4 in each dataset. Converted the class columns in all 4 datasets into categorical. We need to perform Decision tree and SVM techniques and make binary and multi class classification. For Binary class classification I have 3 different models for Genotype, Treatment, Behavior, and multiclass classification model is created for class variable. For making the comparison to be done easily, I have coded tree-based model and SVM model one below the other for each classification problem. For cross validation I have used validation set approach with 80 % train and 20 % test.

While making tree models, I have initially built basic models and observed the confusion matrix. If all the rows are not classified correctly, I have proceeded with pruning and created new models with prune function using the best parameter with size value of lease error. For SVM model, created a SVM classification model using linear kernel and analyzed confusion matrix. If all the rows are not classified correctly, I have used the tune function to identify the cost parameter and used it to build the model again. Same approached is applied for all the 4 classifications for both Trees and SVM models.

Results:

1. **Binary Classification – Genotype:**
   a. Decision Tree:
   - Before Pruning

tree_pred	Control	Ts65Dn
Control	101	9
Ts65Dn	12	94

   - After Pruning

prune_pred	Control	Ts65Dn
Control	104	21
Ts65Dn	9	82

   b. SVM:
   - Before Tuning

```
y_pred    Control Ts65Dn
  Control     113      2
  Ts65Dn        0    101
```

- Tuning Summary to determine cost parameter, which resulted in 0.1

```
- Detailed performance results:
   cost      error dispersion
1 1e-03 0.17711842 0.03989384
2 1e-02 0.05429030 0.02592367
3 1e-01 0.03116814 0.02718946
4 1e+00 0.03809142 0.03391328
5 5e+00 0.04742048 0.02336275
6 1e+01 0.04855654 0.02762846
7 1e+02 0.05780540 0.03258641
```

- After Tuning

```
y_pred_tune Control Ts65Dn
    Control     112      2
    Ts65Dn        1    101
```

Comments: For the binary classification of Genotype, SVM before tuning is performing better than all other models. Tree before pruning is performing worst of all the 4 models.

2. **Binary Classification – Treatment**
   a. Decision Tree:

- Before Pruning

```
tree_pred   Memantine Saline
  Memantine       110      8
  Saline           12     86
```

- After Pruning

```
prune_pred  Memantine Saline
  Memantine       110      8
  Saline           12     86
```

   b. SVM:

- Before Tuning

```
y_pred      Memantine Saline
  Memantine       110      7
  Saline           12     87
```

- Tuning Summary to determine cost parameter, which resulted in 0.1

```
- Detailed performance results:
   cost      error dispersion
1 1e-03 0.30672280 0.06064551
2 1e-02 0.14101844 0.06425566
3 1e-01 0.08672815 0.02874758
4 1e+00 0.08789094 0.03355503
5 5e+00 0.08912056 0.02051723
6 1e+01 0.10764501 0.02056548
7 1e+02 0.11109329 0.03397385
```

- After Tuning

```
y_pred_tune Memantine Saline
  Memantine       110      6
  Saline           12     88
```

Comments: For the binary classification of Treatment, SVM after tuning is performing best of all and SVM before tuning has only 1 misclassification compared to after tuning. Surprisingly both the Tree models has same number of proper classifications.

3. **Binary Classification – Behavior**
   a. Decision Tree:
   - Before Pruning

tree_pred	C/S	S/C
C/S	110	1
S/C	1	104

   - After Pruning

prune_pred	C/S	S/C
C/S	110	1
S/C	1	104

   b. SVM:
   - Before Tuning

y_pred	C/S	S/C
C/S	111	0
S/C	0	105

Comments: For the binary classification of Behavior, SVM before tuning is 100% accurate and classified all the rows correctly and both tree models misclassified 2 rows one in each class.

4. **Multiclass Classification – Class**
   a. Decision Tree:
   - Before Pruning: Had 48 misclassifications

```
> sum(tree_pred!=test_set$class)
[1] 48
> table(tree_pred,test_set$class)

tree_pred c-CS-m c-CS-s c-SC-m c-SC-s t-CS-m t-CS-s t-SC-m t-SC-s
   c-CS-m     26     11      0      0      2      1      0      0
   c-CS-s      5      8      0      0      2      0      0      1
   c-SC-m      0      1     25      2      0      0      2      0
   c-SC-s      0      0      2     21      0      0      0      4
   t-CS-m      6      1      0      0     24      2      0      0
   t-CS-s      1      3      0      0      1     17      0      0
   t-SC-m      0      0      0      1      0      0     26      0
   t-SC-s      0      0      0      0      0      0      0     21
```

   - After Pruning: Had 53 misclassifications

```
> sum(prune_pred!=test_set$class)
[1] 53
> table(prune_pred,test_set$class)

prune_pred c-CS-m c-CS-s c-SC-m c-SC-s t-CS-m t-CS-s t-SC-m t-SC-s
    c-CS-m     27     11      0      0      7      2      0      0
    c-CS-s      5      8      0      0      2      0      0      1
    c-SC-m      0      1     25      2      0      0      3      0
    c-SC-s      0      0      2     21      0      0      0      4
    t-CS-m      5      1      0      0     19      1      0      0
    t-CS-s      1      3      0      0      1     17      0      0
    t-SC-m      0      0      0      1      0      0     25      0
    t-SC-s      0      0      0      0      0      0      0     21
```

   b. SVM:
   - Before Tuning

y_pred	c-CS-m	c-CS-s	c-SC-m	c-SC-s	t-CS-m	t-CS-s	t-SC-m	t-SC-s
c-CS-m	37	0	0	0	0	0	0	0
c-CS-s	1	24	0	0	0	0	0	0
c-SC-m	0	0	27	0	0	0	0	0
c-SC-s	0	0	0	24	0	0	0	0
t-CS-m	0	0	0	0	29	0	0	0
t-CS-s	0	0	0	0	0	20	0	0
t-SC-m	0	0	0	0	0	0	28	0
t-SC-s	0	0	0	0	0	0	0	26

- Tuning Summary to determine cost parameter, which resulted in 0.1

```
- Detailed performance results:
   cost       error dispersion
1 1e-03 0.293918738 0.07101241
2 1e-02 0.043945469 0.02422569
3 1e-01 0.008112804 0.01228181
4 1e+00 0.009262229 0.01197215
5 5e+00 0.009262229 0.01197215
6 1e+01 0.009262229 0.01197215
7 1e+02 0.009262229 0.01197215
```

- After Tuning

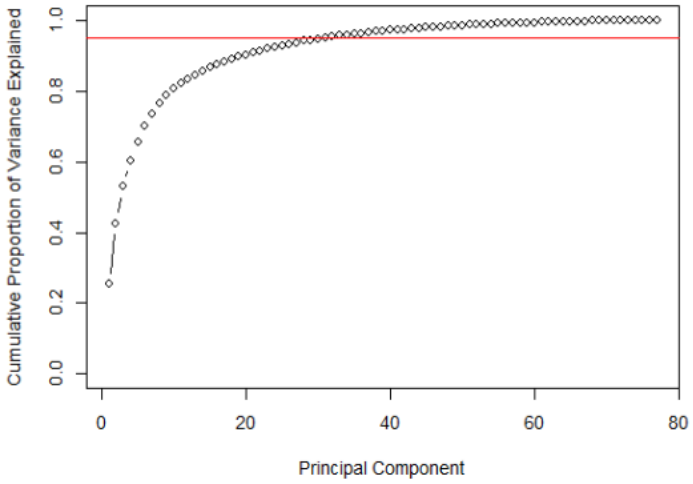
y_pred_tune	c-CS-m	c-CS-s	c-SC-m	c-SC-s	t-CS-m	t-CS-s	t-SC-m	t-SC-s
c-CS-m	38	0	0	0	0	0	0	0
c-CS-s	0	24	0	0	0	0	0	0
c-SC-m	0	0	27	0	0	0	0	0
c-SC-s	0	0	0	24	0	0	0	0
t-CS-m	0	0	0	0	29	0	0	0
t-CS-s	0	0	0	0	0	20	0	0
t-SC-m	0	0	0	0	0	0	28	0
t-SC-s	0	0	0	0	0	0	0	26

Comments: For the binary classification of Class, SVM after tuning had 0 misclassifications and before tuning has 1 misclassification. Both the Tree models has higher number of misclassifications.

Question 1.B

Method:

Using the shuffled_df dataframe, performed principal component analysis. I have used the 77 protein columns and using the prcomp function 77 principal components are created. Plotted the cumulative proportion of variance ratio and check the number of components at 95 percent which was around 30 components. Combined the 30 components with 4 class columns and named it as pca_final. Created 4 dataframes dataset1, dataset2, dataset3, dataset4 having 30 components which represent 77 protein columns, in common and only 1 class column out of 4 in each dataset resulting in 31 columns in each dataframe. Converted the class columns in all 4 datasets into categorical. Like previous question, performed Decision tree and SVM techniques and make binary and multi class classification with similar set up only difference is we have used 30 principal components instead of 77 protein columns.



Results:

1. **Binary Classification – Genotype:**
   a. Decision Tree:
   - Before Pruning

```
tree_pred Control Ts65Dn
  Control      95       9
  Ts65Dn       18      94
```

   - After Pruning

```
prune_pred Control Ts65Dn
   Control      87      17
   Ts65Dn       26      86
```

   b. SVM:
   - Before Tuning

```
y_pred    Control Ts65Dn
  Control     109       3
  Ts65Dn        4     100
```

   - Tuning Summary to determine cost parameter, which resulted in 0.1

```
- Detailed performance results:
   cost       error dispersion
1 1e-03 0.10646886 0.03128770
2 1e-02 0.06357926 0.01953922
3 1e-01 0.06009088 0.02904053
4 1e+00 0.06240310 0.02925865
5 5e+00 0.06586474 0.03205884
6 1e+01 0.06701417 0.03328093
7 1e+02 0.06816359 0.03484099
```

   - After Tuning

```
y_pred_tune Control Ts65Dn
    Control     109       4
    Ts65Dn        4      99
```

Comments: For the binary classification of Genotype, SVM before tuning is performing better than all other models. Tree before after is performing worst of all the 4 models.

2. **Binary Classification – Treatment**
   a. Decision Tree:
   - Before Pruning

```
tree_pred   Memantine Saline
  Memantine       103     14
  Saline           19     80
```

   - After Pruning

```
prune_pred  Memantine Saline
  Memantine        97     17
  Saline           25     77
```

   b. SVM:
   - Before Tuning

```
y_pred      Memantine Saline
  Memantine       104     14
  Saline           18     80
```

   - Tuning Summary to determine cost parameter, which resulted in 0.1

```
- Detailed performance results:
   cost      error dispersion
1 1e-03 0.2881315 0.06405860
2 1e-02 0.1931569 0.06244258
3 1e-01 0.1897086 0.05254339
4 1e+00 0.1919941 0.04941360
5 5e+00 0.1919941 0.05091105
6 1e+01 0.1931435 0.05054833
7 1e+02 0.1919941 0.04944834
```

- After Tuning

```
y_pred_tune Memantine Saline
  Memantine       101      13
  Saline           21      81
```

Comments: For the binary classification of Treatment, SVM before tuning is performing better than all with 32 misclassification and SVM after tuning has 34 misclassifications. Tree model before pruning has 33 misclassifications but pruned tree model is at last with over 40 misclassifications.

3. Binary Classification – Behavior

a. Decision Tree:

- Before Pruning

```
tree_pred C/S S/C
      C/S 103   7
      S/C   8  98
```

- After Pruning

```
prune_pred C/S S/C
       C/S 103   8
       S/C   8  97
```

b. SVM:

- Before Tuning

```
y_pred C/S S/C
   C/S 111   1
   S/C   0 104
```

- Tuning Summary to determine cost parameter, where it was observed to have 0.01 and 0.1 have 0 error. I have opted 0.1 as

```
- Detailed performance results:
   cost       error  dispersion
1 1e-03 0.003475007 0.007832936
2 1e-02 0.000000000 0.000000000
3 1e-01 0.000000000 0.000000000
4 1e+00 0.001162791 0.003677067
5 5e+00 0.001162791 0.003677067
6 1e+01 0.001162791 0.003677067
7 1e+02 0.001162791 0.003677067
```

- After Tuning

```
y_pred_tune C/S S/C
       C/S 111   0
       S/C   0 105
```

Comments: For the binary classification of Behavior, SVM after tuning is 100% accurate and classified all the rows correctly. But there is one misclassification for SVM before tuning. Both the tree models perform almost similar.

4. Multiclass Classification – Class

a. Decision Tree:

- Before Pruning: Had 65 misclassifications

```
> sum(tree_pred!=test_set$class)
[1] 65
> table(tree_pred,test_set$class)

tree_pred c-CS-m c-CS-s c-SC-m c-SC-s t-CS-m t-CS-s t-SC-m t-SC-s
   c-CS-m     18      0      2      0      2      0      0      0
   c-CS-s      5     18      0      0      1      0      0      0
   c-SC-m      1      3     23      0      0      0      5      0
   c-SC-s      1      0      0     19      3      0      2      1
   t-CS-m      8      3      0      0     20      3      2      3
   t-CS-s      0      0      0      0      1     16      0      0
   t-SC-m      5      0      1      5      1      1     18      3
   t-SC-s      0      0      1      0      1      0      1     19
```

- After Pruning: Had 66 misclassifications

```
> sum(prune_pred!=test_set$class)
[1] 66
> table(prune_pred,test_set$class)

prune_pred c-CS-m c-CS-s c-SC-m c-SC-s t-CS-m t-CS-s t-SC-m t-SC-s
    c-CS-m     18      0      2      0      2      0      0      0
    c-CS-s      5     18      0      0      1      0      0      0
    c-SC-m      1      3     23      0      0      0      5      0
    c-SC-s      1      0      0     19      3      0      2      1
    t-CS-m      8      3      0      0     21      3      3      4
    t-CS-s      0      0      0      0      1     16      0      0
    t-SC-m      5      0      1      5      0      1     17      3
    t-SC-s      0      0      1      0      1      0      1     18
```

b. SVM:

- Before Tuning

```
y_pred   c-CS-m c-CS-s c-SC-m c-SC-s t-CS-m t-CS-s t-SC-m t-SC-s
  c-CS-m     37      0      0      0      0      0      0      0
  c-CS-s      1     24      0      0      0      0      0      0
  c-SC-m      0      0     27      0      0      0      0      0
  c-SC-s      0      0      0     24      0      0      1      0
  t-CS-m      0      0      0      0     29      0      0      0
  t-CS-s      0      0      0      0      0     20      0      0
  t-SC-m      0      0      0      0      0      0     27      0
  t-SC-s      0      0      0      0      0      0      0     26
```

- Tuning Summary to determine cost parameter, which resulted in 0.1

```
- Detailed performance results:
   cost       error dispersion
1 1e-02 0.06144079 0.02528870
2 1e-01 0.01390003 0.01063407
3 1e+00 0.02318899 0.01732438
4 5e+00 0.02201283 0.01278912
5 1e+01 0.02201283 0.01278912
6 1e+02 0.02201283 0.01278912
7 1e+03 0.02201283 0.01278912
```

- After Tuning

```
y_pred_tune c-CS-m c-CS-s c-SC-m c-SC-s t-CS-m t-CS-s t-SC-m t-SC-s
     c-CS-m     37      1      0      0      0      0      0      0
     c-CS-s      1     23      0      0      0      0      0      0
     c-SC-m      0      0     27      0      0      0      0      0
     c-SC-s      0      0      0     24      0      0      1      0
     t-CS-m      0      0      0      0     29      0      0      0
     t-CS-s      0      0      0      0      0     20      0      0
     t-SC-m      0      0      0      0      0      0     27      0
     t-SC-s      0      0      0      0      0      0      0     26
```

Comments: For the binary classification of Class, SVM before tuning had 2 misclassifications and before tuning has 3 misclassifications. Both the Tree models has higher number of misclassifications.

Question 1.C

Method:

Using the shuffled_df dataframe, created 4 dataframes dataset1, dataset2, dataset3, dataset4 having 77 protein columns in common and only 1 class column out of 4 in each dataset. Converted the class columns in all 4 datasets into numerical values and changed type to factor / categorical. We need to perform Bagging, Random Forest and Boosting techniques and make binary and multi class classification. For Binary class classification I have 3 different models for Genotype, Treatment, Behavior, and multiclass classification model is created for class variable. For making the comparison to be done easily, I have coded three models one below the other.

Results:

1. **Binary Classification – Genotype: Control – 0, Ts65Dn-1**
   a. Bagging:

```
bag_pred   0   1
       0 119   3
       1   1  93
```

   b. Random Forest:

```
rf_pred   0   1
      0 120   1
      1   0  95
```

   c. Boosting:

```
boost_pred   0   1
         0 119   0
         1   1  96
```

Comments: For the binary classification of Genotype, random forest and boosting performs the best with only 1 misclassification but bagging had 4 misclassifications.

2. **Binary Classification – Treatment: Memantine – 0, Saline - 1**
   a. Bagging:

```
bag_pred   0   1
       0 110   1
       1   4 101
```

   b. Random Forest:

```
rf_pred   0   1
      0 114   0
      1   0 102
```

   c. Boosting:

```
boost_pred   0   1
         0 113   4
         1   1  98
```

Comments: For the binary classification of Treatment, Random Forest outperforms bagging and boosting with 100 percent accuracy where as bagging and boosting has 5 misclassifications.

3. **Binary Classification – Behavior: C/S – 0, S/C - 1**
   a. Bagging:

bag_pred	0	1
0	116	0
1	0	100

   b. Random Forest:

rf_pred	0	1
0	116	0
1	0	100

   c. Boosting:

boost_pred	0	1
0	116	0
1	0	100

Comments: For the binary classification of Behavior, all 3 performed well with 0 misclassifications.

4. **Multiclass Classification – Class: c-CS-m - 0, c-CS-s - 1, c-SC-m - 2, c-SC-s -3, t-CS-m - 4, t-CS-s - 5, t-SC-m - 6, t-SC-s - 7**
   a. Bagging:

bag_pred	0	1	2	3	4	5	6	7
0	37	0	0	0	0	1	0	0
1	0	29	0	0	0	0	0	0
2	0	0	24	0	0	0	0	0
3	0	0	0	26	0	0	1	0
4	0	0	0	0	26	0	0	0
5	0	0	0	0	0	23	0	0
6	0	0	0	0	0	0	26	0
7	0	0	0	0	0	0	0	23

   b. Random Forest:

rf_pred	0	1	2	3	4	5	6	7
0	36	0	0	0	0	0	0	0
1	1	29	0	0	0	0	0	0
2	0	0	24	0	0	0	0	0
3	0	0	0	26	0	0	0	0
4	0	0	0	0	26	1	0	0
5	0	0	0	0	0	23	0	0
6	0	0	0	0	0	0	27	0
7	0	0	0	0	0	0	0	23

   c. Boosting:

boost_pred	1	2	3	4	5	6	7	8
1	37	0	0	0	0	0	0	0
2	0	29	0	0	1	1	0	0
3	0	0	24	0	0	0	0	0
4	0	0	0	26	0	0	0	0
5	0	0	0	0	25	4	0	0
6	0	0	0	0	0	19	0	0
7	0	0	0	0	0	0	27	0
8	0	0	0	0	0	0	0	23

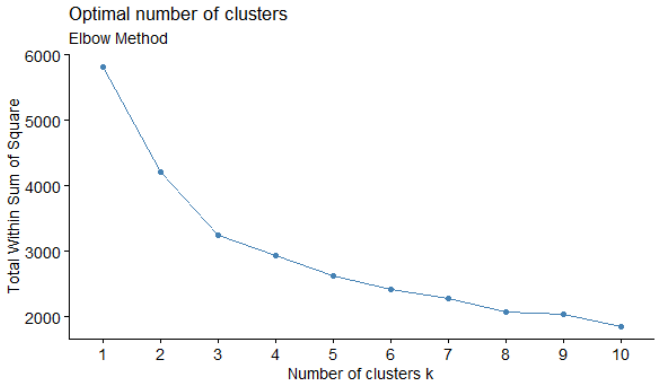
Comments: For the binary classification of Class, bagging and random forest models had almost the same accuracy with 2 misclassifications each. Boosting had 6 misclassifications.

Question 2

Method:

Using the shuffled_df, I have extracted the 77 protein columns, and used that dataframe as input to the clustering models. It was asked to perform K-Means and Hierarchical clustering's.

Firstly, for the K-Means clustering, using elbow methos I have tried to predict the number of clusters to be used for the model. I have considered 3 clusters based on the elbow plot and below is the corresponding plot.

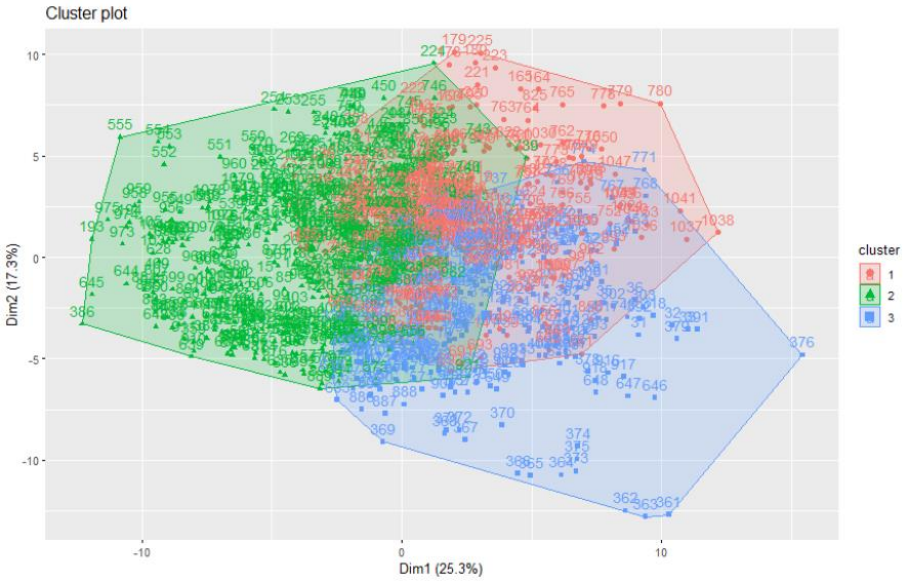


For Hierarchical clustering, I have performed complete linkage clustering as it looked more evenly distributed when compared to single linkage or average linkage. We decide the number of clusters based on the plot of the cluste.

Results:

K Means Clustering:

Below image shows the 3 clusters generated based on K-Means Clustering.



Below table shows how the categorical columns are separated into each cluster.

		Cluster1	Cluster2	Cluster3
Genotype	Control	144	233	193
	Ts65Dn	189	237	84
Treatment	Memantine	230	212	128
	Saline	103	258	149
Behaviour	C/S	62	246	217
	S/C	271	224	60
Class	c-CS-m	13	55	82
	c-CS-s	9	57	69
	c-SC-m	101	49	0
	c-SC-s	21	72	42
	t-CS-m	31	72	32
	t-CS-s	9	62	34
	t-SC-m	85	36	14
	t-SC-s	64	67	4

Cluster 1 – Clustered data of the mice, which are majorly shock context and treated with memantine which are mostly control.
Cluster 2 – Balanced cluster where all the columns have almost close values, except for shock context and memantine treated mice with down syndrome are too low.
Cluster 3 – Clustered data of mice which are majorly control type, treated with Saline and context shock behavior.

Hierarchical Clustering:

A line drawn at 6 units of y axis produced 7 clusters which I felt would be a better number of clusters as the clusters on the left looks evenly distributed compared to right side.