CS7641 Homework 3 – Jayden Cho

Table of Contents

0. Introduction

This report conveys part of unsupervised machine learning analysis and dimensionality reduction with R Studio. R is a free software environment generally used for computing and statistical analysis. Libraries used and more information about the software are included in the README file.

This will introduce two different datasets for comparison in different algorithms' accuracy of how well to predict their response variables. Also, combinations of those algorithms will be introduced to depict whether prediction accuracy increases or decreases if used together.

1. Problem Set 1: Wine Data

1.1 Data Description

The first problem set to address is wine data set. These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. The 13 attributes include one response variable and 12 predictors. The alcohol, the response variable, is represented with 1,2, or 3, in which each number represents type of wine. The other attributes are chemical characteristics of ingredients and physical characteristics of the wine itself. The dataset has 178 observations, which is a far greater number than the number of attributes, so that this dataset is suitable to be used in classification and clustering algorithms. The goal is to correctly predict the type of wine given attributes.

```
#  Attribute
   --------------------------------------------
   1. Alcohol                            1,2 or 3
   2. Malic Acid
   3. Ash
   4. Alkalinity of ash
   5. Magnesium
   6. Total Phenols
   7. Flavonoids
   8. Non-flavonoid Phenols
   9. Proanthocyanins
  10. Color Intensity
  11. Hue
  12. OD280/OD315 of Diluted Wines
  13. Proline
```

1.2 Algorithms

In this problem set, 3 types of algorithms or combinations of algorithms are demonstrated: clustering, dimensionality reduction, and clustering on dimensionality reduction.

1.2.1 Clustering

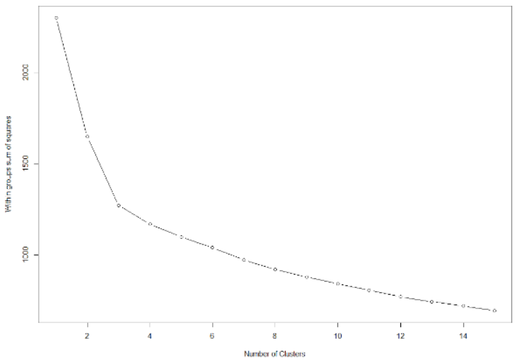
Clustering is part of unsupervised machine learning analysis, where the task is grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters). For more information of cluster analysis as a whole and tutorials: https://home.deib.polimi.it/matteucc/Clustering/tutorial_html/. For this report, two clustering algorithms are used: K-Means Clustering and Expectation Maximization.

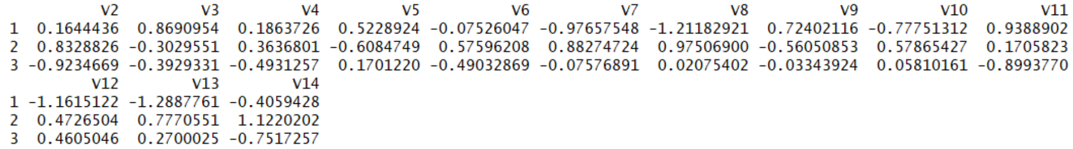
1.2.1.1 K-Means clustering

Its purpose is to partition a set of vectors into K groups that cluster around common mean vector. This can also be thought as approximating the input each of the input vector with one of the means, so the clustering process finds, in principle, the best dictionary or codebook to vector quantize the data.

For this dataset, the optimal number of clusters should be definite before running the actual k-means algorithm since the parameter for the number of clusters is user-defined. 15 cases are tested, where each case is assigned with different number of clusters and its corresponding within groups sum of squares.

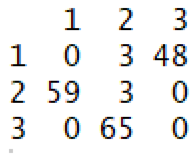
This plot of the within groups sum of squares by number of clusters extracted can help determine the appropriate number of clusters. Since the elbow is located at 3 number of clusters, it is determined that 3 is the optimal number. Then, k-means algorithm is then run without the response variable and resulted in 3 different clusters. The centers of those clusters are in multidimensional coordinates as expected with 13 predictors:





```
          V2         V3         V4         V5          V6          V7          V8          V9         V10        V11
1  0.1644436  0.8690954  0.1863726  0.5228924 -0.07526047 -0.97657548 -1.21182921  0.72402116 -0.77751312  0.9388902
2  0.8328826 -0.3029551  0.3636801 -0.6084749  0.57596208  0.88274724  0.97506900 -0.56050853  0.57865427  0.1705823
3 -0.9234669 -0.3929331 -0.4931257  0.1701220 -0.49032869 -0.07576891  0.02075402 -0.03343924  0.05810161 -0.8993770
         V12        V13        V14
1 -1.1615122 -1.2887761 -0.4059428
2  0.4726504  0.7770551  1.1220202
3  0.4605046  0.2700025 -0.7517257
```

The total sum of squares is 2301 and within-cluster sum of squares per cluster is: 326.3537, 385.6983, and 558.6971. The sizes of those 3 clusters are 51, 62, and 65 observations. The predicted cluster assignments and the actual classification are compared:



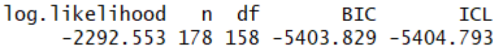
```
    1  2  3
1   0  3 48
2  59  3  0
3   0 65  0
```

The clusters do not line up with the labels because the labels are not given to the algorithm when training. Although, only 6 out of 178 observations are wrongfully clustered, which is about 3.37%. This algorithm can be further improved with optimal set of parameters, such as number of random sets at initialization, type of algorithm, and so on.

1.2.1.2 Expectation Maximization

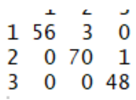
Expectation maximization is also clustering algorithm, where an iterative method is conducted to find maximum likelihood estimates of parameters in statistical models that depend on unobserved latent variables. This algorithm is useful when the equations cannot be solved directly. Those models typically include latent variables and unknown parameters, which are suitable conditions to be used in unsupervised machine learning techniques. For this problem set, 'Mclust' package is used to fit the data to an expectation maximization model. The only parameter that is defined is the number of clusters, which is already determined by the plot above as 3.

The statistical components of the model are:

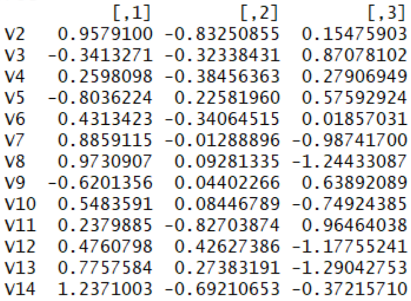


```
 log.likelihood   n  df       BIC       ICL
      -2292.553 178 158 -5403.829 -5404.793
```

The observations are grouped into 3 clusters with these sizes: 56, 73, and 49. The tables for the centers of the clusters and comparison with the actual labels are shown below. These 3 centers, again, are in 13 dimensions due to the number of predictors.



```
   1  2  3
1 56  3  0
2  0 70  1
3  0  0 48
```



```
          [,1]        [,2]        [,3]
V2   0.9579100 -0.83250855  0.15475903
V3  -0.3413271 -0.32338431  0.87078102
V4   0.2598098 -0.38456363  0.27906949
V5  -0.8036224  0.22581960  0.57592924
V6   0.4313423 -0.34064515  0.01857031
V7   0.8859115 -0.01288896 -0.98741700
V8   0.9730907  0.09281335 -1.24433087
V9  -0.6201356  0.04402266  0.63892089
V10  0.5483591  0.08446789 -0.74924385
V11  0.2379885 -0.82703874  0.96464038
V12  0.4760798  0.42627386 -1.17755241
V13  0.7757584  0.27383191 -1.29042753
V14  1.2371003 -0.69210653 -0.37215710
```

The table on the right shows that the labels naturally line up with the original classifications and that only 4 of the observations are in the wrong clusters, resulted in 2.25% error rate. This algorithm has a better outcome than the k-means algorithm. Also, it may be possible to reduce the error rate by more closely determining the optimal set of parameters.

1.2.2 Dimensionality Reduction

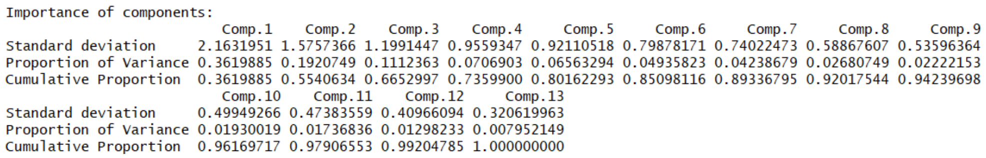
Dimension Reduction refers to the process of converting a set of data having vast dimensions into data with lesser dimensions ensuring that it conveys similar information concisely. These techniques are typically used while solving machine learning problems to obtain better features for classification, regression, clustering, etc. tasks.

In this analysis, 4 different algorithms are applied to the wine dataset: Principal Component Analysis (PCA), Independent Component Analysis (ICA), Randomized Projections (RP), and __.

1.2.2.1 Principal Component Analysis (PCA)

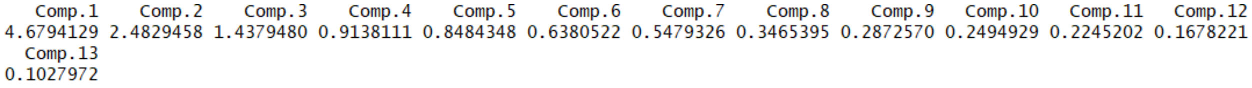
Principal component analysis (PCA) simplifies the complexity in high-dimensional data while retaining trends and patterns. It does this by transforming the data into fewer dimensions, which act as summaries of features.

With the default 'princomp' method in R, the predictors in the wine dataset is converted to principal components, and the distributions are as follow:

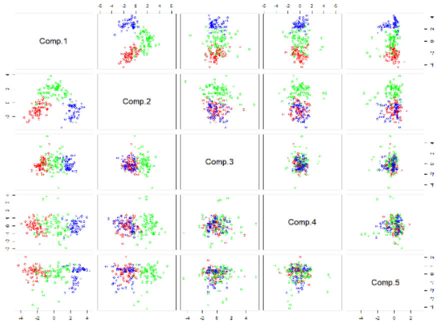


```
Importance of components:
                          Comp.1    Comp.2    Comp.3    Comp.4     Comp.5     Comp.6     Comp.7     Comp.8     Comp.9
Standard deviation     2.1631951 1.5757366 1.1991447 0.9559347 0.92110518 0.79878171 0.74022473 0.58867607 0.53596364
Proportion of Variance 0.3619885 0.1920749 0.1112363 0.0706903 0.06563294 0.04935823 0.04238679 0.02680749 0.02222153
Cumulative Proportion  0.3619885 0.5540634 0.6652997 0.7359900 0.80162293 0.85098116 0.89336795 0.92017544 0.94239698
                          Comp.10    Comp.11    Comp.12     Comp.13
Standard deviation     0.49949266 0.47383559 0.40966094 0.320619963
Proportion of Variance 0.01930019 0.01736836 0.01298233 0.007952149
Cumulative Proportion  0.96169717 0.97906553 0.99204785 1.000000000
```

The components are aligned by their importance, first component being the most important regarding the data. The proportion of variance conveys how much the variance in the data can be interpreted with the component. The first component explains about 36.20% of the data, and so on. The eigenvalues for each component are:



```
   Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7    Comp.8    Comp.9   Comp.10   Comp.11   Comp.12 
4.6794129 2.4829458 1.4379480 0.9138111 0.8484348 0.6380522 0.5479326 0.3465395 0.2872570 0.2494929 0.2245202 0.1678221 
  Comp.13 
0.1027972 
```

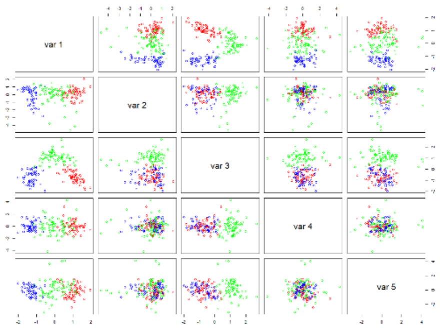


The following plot shows the pairwise plots of the first 5 principal components (PCs). Each plot can be seen as a transformation in a 2-dimensional space where the 2 components are the new coordinate system. The projection along the first PC separates the three classes quite well, the best result seems to be a combination of 1st and 4th component. Although PCA is very useful in finding the components with the highest variation, it does not always mean that the component is useful for separation in that order.

1.2.2.2 Independent Component Analysis (ICA)

The purpose of ICA is to find independent components in the data. In contrast to PCA you do not automatically get the same number of components as you have dimensions. Independent components also do not have to be orthogonal and they are not ranked (there is no "most independent component"). ICA is an extension to principal component analysis and factor analysis. ICA is a much more powerful technique, however, capable of finding the underlying factors or sources when these classic methods fail completely.

In terms of this dataset, it is reduced to 5 components and the pairwise plot is as following.



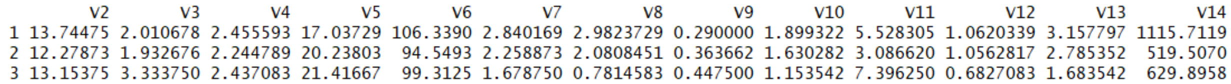
This plot tells us that the combination of variable 1 and 4 can interpret or cluster the data most correctly. The kurtosis of each component is -1.262, 4.405, -1.010, 3.061, and 3.007, respectively, and the variables 4 and 5 are most close to normal distribution.

1.2.2.3 Randomized Projections (RP)

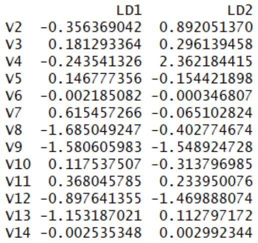
Random projection is also a tool for representing high-dimensional data in a low-dimensional feature space, typically for data visualization or methods that rely on fast computation of pairwise distances, like nearest neighbors searching and nonparametric clustering. The algorithm returns the original dataset, rotation, and the resulting dimension-reduced matrix.

1.2.2.4 Linear Discriminant Analysis (LDA)

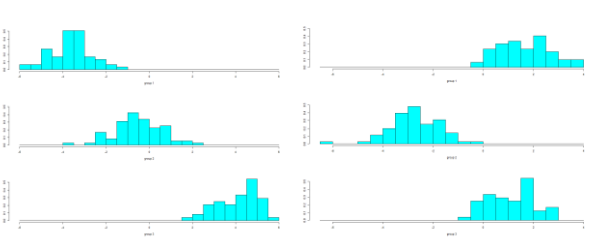
LDA is a well-established machine learning technique for predicting categories. Its main advantages, compared to other classification algorithms such as neural networks and random forests, are that the model is interpretable and that prediction is easy. When the LDA is calculated, here is the means and the actual components plus histograms of predicted values/classes by the first discriminant function and the second discriminant function, as well as the scatterplot of the best two discriminant functions are:

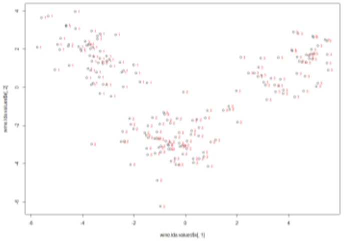


```
        V2       V3       V4       V5       V6       V7        V8       V9      V10      V11       V12      V13       V14
1 13.74475 2.010678 2.455593 17.03729 106.3390 2.840169 2.9823729 0.290000 1.899322 5.528305 1.0620339 3.157797 1115.7119
2 12.27873 1.932676 2.244789 20.23803  94.5493 2.258873 2.0808451 0.363662 1.630282 3.086620 1.0562817 2.785352  519.5070
3 13.15375 3.333750 2.437083 21.41667  99.3125 1.678750 0.7814583 0.447500 1.153542 7.396250 0.6827083 1.683542  629.8958
```



```
             LD1          LD2
V2  -0.356369042  0.892051370
V3   0.181293364  0.296139458
V4  -0.243541326  2.362184415
V5   0.146777356 -0.154421898
V6  -0.002185082 -0.000346807
V7   0.615457266 -0.065102824
V8  -1.685049247 -0.402774674
V9  -1.580605983 -1.548924728
V10  0.117537507 -0.313796985
V11  0.368045785  0.233950076
V12 -0.897641355 -1.469888074
V13 -1.153187021  0.112797172
V14 -0.002535348  0.002992344
```



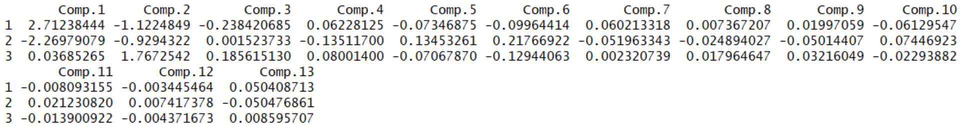


1.2.3 Clustering on Dimensionality Reduction

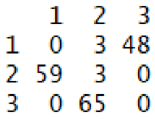
In this section, the clustering algorithms above are applied again to the newly projected data after the dimensionality reduction algorithms are placed on the original data, and the results are compared to see if whether they improved.

1.2.3.1 K-Means Clustering on PCA

The k-means algorithm apparently resulted in different cluster centers based on the newly projected data by PCA algorithm:



```
       Comp.1     Comp.2       Comp.3      Comp.4      Comp.5      Comp.6       Comp.7       Comp.8      Comp.9     Comp.10
1  2.71238444 -1.1224849 -0.238420685  0.06228125 -0.07346875 -0.09964414  0.060213318  0.007367207  0.01997059 -0.06129547
2 -2.26979079 -0.9294322  0.001523733 -0.13511700  0.13453261  0.21766922 -0.051963343 -0.024894027 -0.05014407  0.07446923
3  0.03685265  1.7672542  0.185615130  0.08001400 -0.07067870 -0.12944063  0.002320739  0.017964647  0.03216049 -0.02293882
       Comp.11      Comp.12      Comp.13
1 -0.008093155 -0.003445464  0.050408713
2  0.021230820  0.007417378 -0.050476861
3 -0.013900922 -0.004371673  0.008595707
```



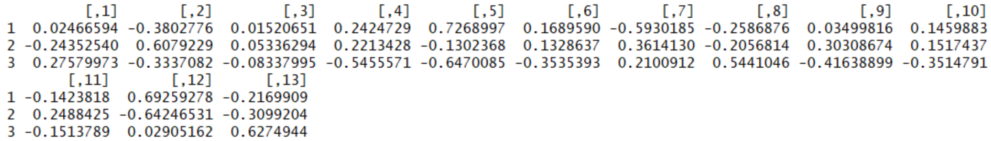
```
   1  2  3
1  0  3 48
2 59  3  0
3  0 65  0
```

However, the total sum of squares, 2301, and within-cluster sums of squares per cluster, 326.3537, 385.6983, and 558.6971 remain the same as well as the comparison between predicted clusters and actual labels:

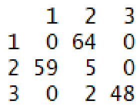
Additionally, the error rate surely remains the same as 3.37%.

1.2.3.2 K-Means Clustering on ICA

After performing k-means clustering algorithm on the newly projected data by ICA, the centers are:



```
        [,1]       [,2]        [,3]       [,4]       [,5]       [,6]       [,7]       [,8]        [,9]      [,10]
1  0.02466594 -0.3802776  0.01520651  0.2424729  0.7268997  0.1689590 -0.5930185 -0.2586876  0.03499816  0.1459883
2 -0.24352540  0.6079229  0.05336294  0.2213428 -0.1302368  0.1328637  0.3614130 -0.2056814  0.30308674  0.1517437
3  0.27579973 -0.3337082 -0.08337995 -0.5455571 -0.6470085 -0.3535393  0.2100912  0.5441046 -0.41638899 -0.3514791
       [,11]       [,12]      [,13]
1 -0.1423818  0.69259278 -0.2169909
2  0.2488425 -0.64246531 -0.3099204
3 -0.1513789  0.02905162  0.6274944
```

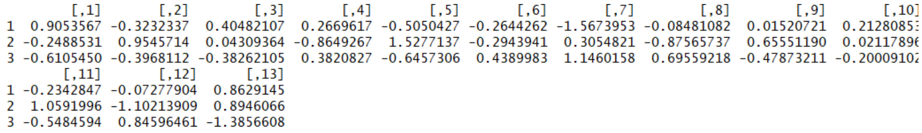


```
   1  2  3
1  0 64  0
2 59  5  0
3  0  2 48
```

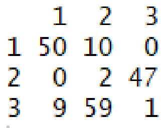
The total sum of squares is little higher at 2314 and the within-cluster sums of squares per cluster are also higher at 475.2564, 933.6355, and 600.4384. The confusion matrix is shown on the left. The error rate increased by a little as the model wrongfully cluster 7 observations: 3.393%.

1.2.3.3 K-Means Clustering on RP

K-means clustering on RP resulted in different set of centers with the new data projected by random projections:



```
        [,1]       [,2]        [,3]       [,4]       [,5]       [,6]       [,7]        [,8]        [,9]      [,10]
1  0.9053567 -0.3232337  0.40482107  0.2669617 -0.5050427 -0.2644262 -1.5673953 -0.08481082  0.01520721  0.2128085
2 -0.2488531  0.9545714  0.04309364 -0.8649267  1.5277137 -0.2943941  0.3054821 -0.87565737  0.65551190  0.0211789
3 -0.6105450 -0.3968112 -0.38262105  0.3820827 -0.6457306  0.4389983  1.1460158  0.69559218 -0.47873211 -0.2000910
       [,11]       [,12]      [,13]
1 -0.2342847 -0.07277904  0.8629145
2  1.0591996 -1.10213909  0.8946066
3 -0.5484594  0.84596461 -1.3856608
```

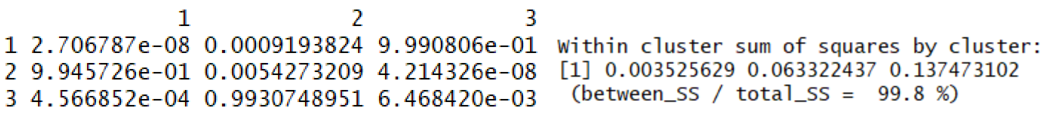


```
   1  2  3
1 50 10  0
2  0  2 47
3  9 59  1
```

The total sum of squares is lower at 2402.201 and the within-cluster sums of squares per cluster are 438.7331, 320.3926, and 491.1609. The confusion matrix shows that the error is 12.36%. These centers, sum of squares, confusion matrix, cluster assignments, etc. are subject to change on a different run of random projections.

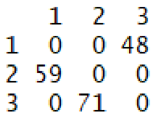
1.2.3.4 K-Means Clustering on LDA

K-Means clustering on LDA posterior results in perfect clustering with 100% accuracy:



```
             1            2            3
1 2.706787e-08 0.0009193824 9.990806e-01
2 9.945726e-01 0.0054273209 4.214326e-08
3 4.566852e-04 0.9930748951 6.468420e-03
```

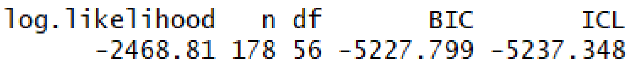
```
Within cluster sum of squares by cluster:
[1] 0.003525629 0.063322437 0.137473102
 (between_SS / total_SS =  99.8 %)
```



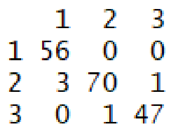
```
   1  2  3
1  0  0 48
2 59  0  0
3  0 71  0
```

1.2.3.5 EM Clustering on PCA

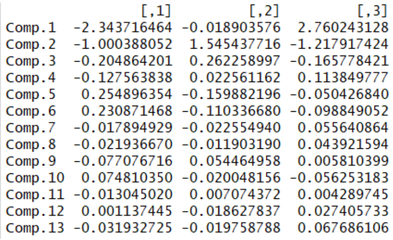
When EM clustering algorithm is performed on the projected data by PCA, log likelihood decreased, and BIC increased.



```
log.likelihood   n df       BIC       ICL
      -2468.81 178 56 -5227.799 -5237.348
```



```
   1  2  3
1 56  0  0
2  3 70  1
3  0  1 47
```

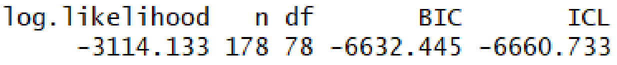


```
                [,1]         [,2]         [,3]
Comp.1  -2.343716464 -0.018903576  2.760243128
Comp.2  -1.000388052  1.545437716 -1.217917424
Comp.3  -0.204864201  0.262258997 -0.165778421
Comp.4  -0.127563838  0.022561162  0.113849777
Comp.5   0.254896354 -0.159882196 -0.050426840
Comp.6   0.230871468 -0.110336680 -0.098849052
Comp.7  -0.017894929 -0.022554940  0.055640864
Comp.8  -0.021936670 -0.011903190  0.043921594
Comp.9  -0.077076716  0.054464958  0.005810399
Comp.10  0.074810350 -0.020048156 -0.056253183
Comp.11 -0.013045020  0.007074372  0.004289745
Comp.12  0.001137445 -0.018627837  0.027405733
Comp.13 -0.031932725 -0.019758788  0.067686106
```

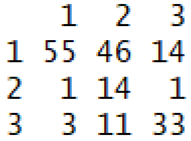
The EM algorithm resulted in these centers and confusion matrix to compare. According to the confusion matrix, only 5 out of 178 observations are in wrong clusters: 2.81%. However, this error rate slightly increased compared to the original EM results.

1.2.3.6 EM Clustering on ICA

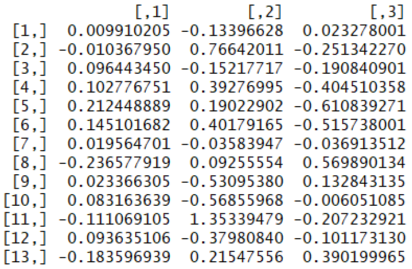
The log likelihood and BIC further decreased when EM is performed on ICA.



```
log.likelihood   n df       BIC       ICL
     -3114.133 178 78 -6632.445 -6660.733
```



```
   1  2  3
1 55 46 14
2  1 14  1
3  3 11 33
```

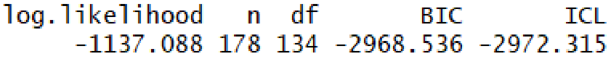


```
              [,1]        [,2]         [,3]
 [1,]  0.009910205 -0.13396628  0.023278001
 [2,] -0.010367950  0.76642011 -0.251342270
 [3,]  0.096443450 -0.15217717 -0.190840901
 [4,]  0.102776751  0.39276995 -0.404510358
 [5,]  0.212448889  0.19022902 -0.610839271
 [6,]  0.145101682  0.40179165 -0.515738001
 [7,]  0.019564701 -0.03583947 -0.036913512
 [8,] -0.236577919  0.09255554  0.569890134
 [9,]  0.023366305 -0.53095380  0.132843135
[10,]  0.083163639 -0.56855968 -0.006051085
[11,] -0.111069105  1.35339479 -0.207232921
[12,]  0.093635106 -0.37980840 -0.101173130
[13,] -0.183596939  0.21547556  0.390199965
```

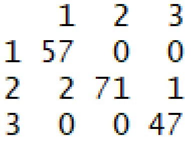
However, the error rate increased significantly to 16.85% primarily due to simplicity of the problem.

1.2.3.7 EM Clustering on RP

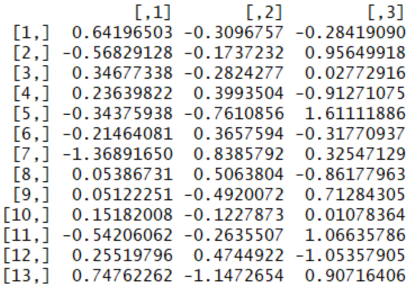
After EM is performed on data projected by RP, the statistics, the cluster centers, and the confusion matrix are as shown below:



```
log.likelihood   n  df       BIC       ICL
     -1137.088 178 134 -2968.536 -2972.315
```



```
   1  2  3
1 57  0  0
2  2 71  1
3  0  0 47
```

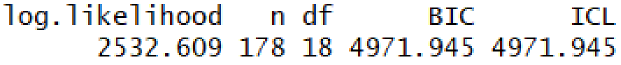


```
             [,1]       [,2]        [,3]
 [1,]  0.64196503 -0.3096757 -0.28419090
 [2,] -0.56829128 -0.1737232  0.95649918
 [3,]  0.34677338 -0.2824277  0.02772916
 [4,]  0.23639822  0.3993504 -0.91271075
 [5,] -0.34375938 -0.7610856  1.61111886
 [6,] -0.21464081  0.3657594 -0.31770937
 [7,] -1.36891650  0.8385792  0.32547129
 [8,]  0.05386731  0.5063804 -0.86177963
 [9,]  0.05122251 -0.4920072  0.71284305
[10,]  0.15182008 -0.1227873  0.01078364
[11,] -0.54206062 -0.2635507  1.06635786
[12,]  0.25519796  0.4744922 -1.05357905
[13,]  0.74762262 -1.1472654  0.90716406
```

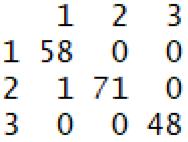
The log likelihood value increased, and BIC value decreased. The centers are different from the original EM clusters and the confusion matrix indicates that only 3 observations are in wrong clusters, 1.69% error.

1.2.3.8 EM Clustering on LDA

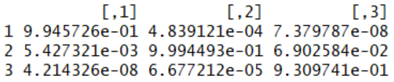
When EM is performed on LDA, however, one is misclassified:



```
log.likelihood   n df      BIC      ICL
      2532.609 178 18 4971.945 4971.945
```



```
   1  2  3
1 58  0  0
2  1 71  0
3  0  0 48
```



```
          [,1]         [,2]         [,3]
1 9.945726e-01 4.839121e-04 7.379787e-08
2 5.427321e-03 9.994493e-01 6.902584e-02
3 4.214326e-08 6.677212e-05 9.309741e-01
```

The log likelihood increased substantially as well as the BIC, 0.562% error.

1.3 Conclusion

The purpose of this problem was to correctly identify the types of wine given chemical and phycial characteristics of various wine. The lowest error rate, 0%, could be achieved when the k-means clustering algorithm was used on the newly projected data by the linear discriminant analysis algorithm due to the problem domain and problem simplicity, followed by EM on LDA, EM on RP, and EM only.

2. Problem Set 2: Breast Cancer Data

2.1 Data Description

The second dataset is regarding whether a patient has a breast cancer or not. For classification purpose, the dataset is divided into 10 attributes and a response variable. 683 data points without missing data are far greater than the number of attributes, and that makes this dataset quite suitable for any classification problem. The goal here is to correctly classify if a patient has breast cancer cells or not by only using characteristics of the cells. Here is more information about the attributes.

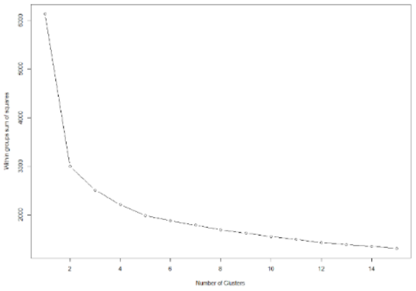
```
#  Attribute                     Domain
   -- -----------------------------------------
   1. Sample code number            id number
   2. Clump Thickness               1 - 10
   3. Uniformity of Cell Size       1 - 10
   4. Uniformity of Cell Shape      1 - 10
   5. Marginal Adhesion             1 - 10
   6. Single Epithelial Cell Size   1 - 10
   7. Bare Nuclei                   1 - 10
   8. Bland Chromatin               1 - 10
   9. Normal Nucleoli               1 - 10
  10. Mitoses                       1 - 10
  11. Class:                    2 for benign, 4 for malignant
```

Bare nuclei correspond to naked nuclei, which is typically seen in cell degeneration. More information can be found at: https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.names.

2.2 Algorithms

In this section, various algorithms and combinations of algorithms are tested as above, and effects on neural network after the combinations are measured to reach the lowest error possible. In the 2.2.1, 2.2.2, and 2.2.3 sections, explanations are skipped since concepts in those sections overlap with the problem set 1 above.

2.2.1 Clustering

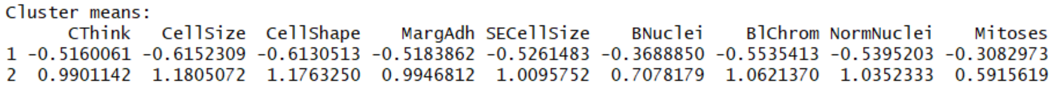


The optimal number of clusters is found by using similar method like above:

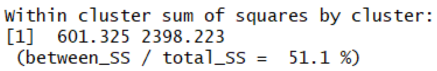
Since the elbow is at 2, the number of clusters in all the clustering algorithms are set to 2.

2.2.1.1 K-Means Clustering

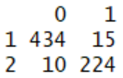
K-Means algorithm on this dataset results in:



```
Cluster means:
      CThink   CellSize  CellShape   MargAdh SECellSize    BNuclei    BlChrom NormNuclei    Mitoses
1 -0.5160061 -0.6152309 -0.6130513 -0.5183862 -0.5261483 -0.3688850 -0.5535413 -0.5395203 -0.3082973
2  0.9901142  1.1805072  1.1763250  0.9946812  1.0095752  0.7078179  1.0621370  1.0352333  0.5915619
```



```
Within cluster sum of squares by cluster:
[1]  601.325 2398.223
 (between_SS / total_SS =  51.1 %)
```

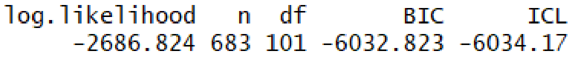


```
    0   1
1 434  15
2  10 224
```

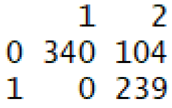
Therefore, 25 out of 683 observations are placed in wrong clusters; 3.66% error rate.

2.2.1.2 Expectation Maximization Clustering

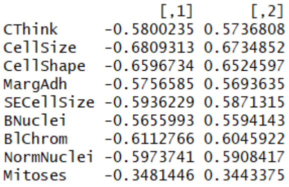
The EM clustering algorithm model is fit to this data and produced this result:



```
log.likelihood   n  df       BIC      ICL
     -2686.824 683 101 -6032.823 -6034.17
```



```
    1   2
0 340 104
1   0 239
```



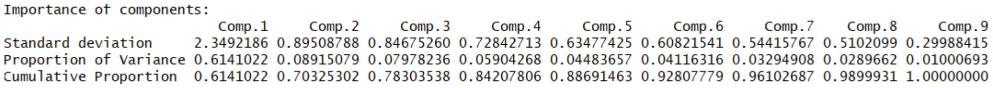
```
                  [,1]      [,2]
CThink      -0.5800235 0.5736808
CellSize    -0.6809313 0.6734852
CellShape   -0.6596734 0.6524597
MargAdh     -0.5756585 0.5693635
SECellSize  -0.5936229 0.5871315
BNuclei     -0.5655993 0.5594143
BlChrom     -0.6112766 0.6045922
NormNuclei  -0.5973741 0.5908417
Mitoses     -0.3481446 0.3443375
```

It resulted in predicting 104 observations in different clusters; 15.23% error rate.

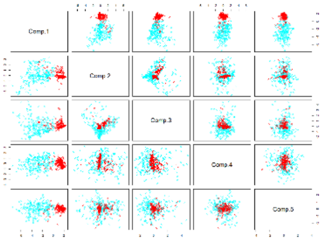
2.2.2 Dimensionality Reduction

2.2.2.1 Principal Component Analysis (PCA)

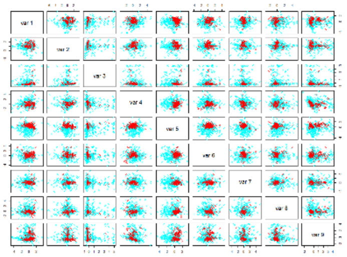
The PCA algorithm gives:



```
Importance of components:
                          Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6     Comp.7    Comp.8     Comp.9
Standard deviation     2.3492186 0.89508788 0.84675260 0.72842713 0.63477425 0.60821541 0.54415767 0.5102099 0.29988415
Proportion of Variance 0.6141022 0.08915079 0.07978236 0.05904268 0.04483657 0.04116316 0.03294908 0.0289662 0.01000693
Cumulative Proportion  0.6141022 0.70325302 0.78303538 0.84207806 0.88691463 0.92807779 0.96102687 0.9899931 1.00000000
```



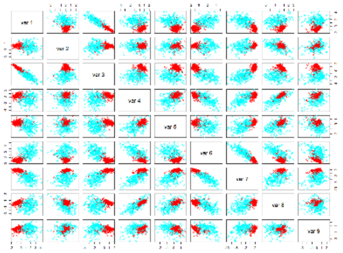
The first component explains 61.41% of the data, and the combination of the first and the fourth components seems to explain the data best according to the plot.



2.2.2.2 Independent Component Analysis (ICA)

After running ICA algorithm on this data, the plot of two-way combinations of components is shown next:

Only few of these components seem useful, which are the fourth and the fifth components. It seems like they also are able to separate data on its own.



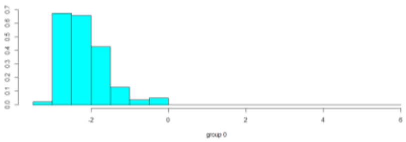
2.2.2.3 Randomized Projections (RP)

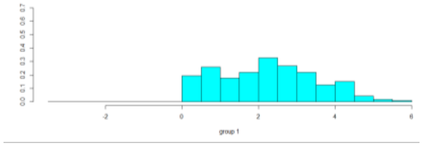
As mentioned above, this algorithm is built from scratch, and therefore, it is difficult to depict meaningful visualization. The algorithm still returns the original dataset, rotation, and the resulting dimension-reduced matrix.

From the projections above, it looks like most of the components cluster the observations correctly, such as variable 3 and 4.

2.2.2.4 Linear Discriminant Analysis (LDA)

After LDA algorithm is run on the data, it is reduced to one dimension. The classification histogram rendered by the linear discriminant component looks working well:

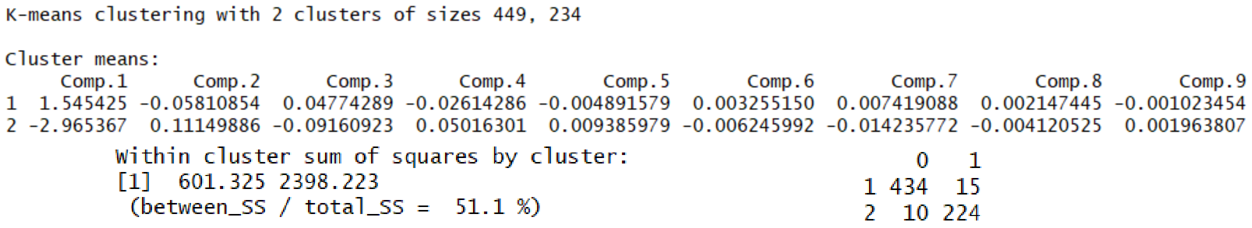




2.2.3 Clustering on Dimensionality Reduction

2.2.3.1 K-Means Clustering on PCA

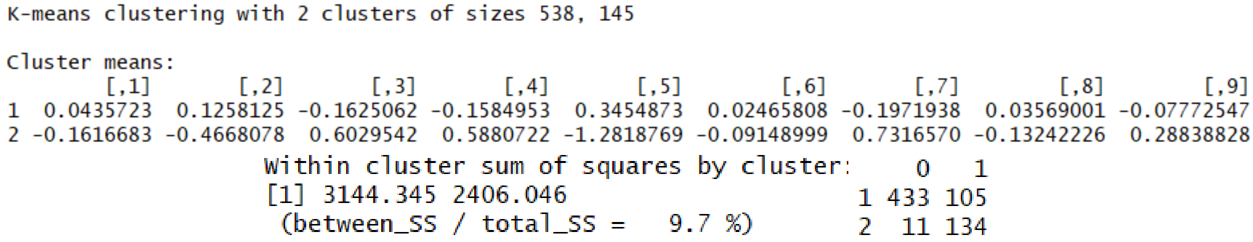
K-Means clustering algorithm is performed on PCA projections:



```
K-means clustering with 2 clusters of sizes 449, 234

Cluster means:
     Comp.1      Comp.2      Comp.3      Comp.4       Comp.5       Comp.6       Comp.7       Comp.8       Comp.9
1  1.545425 -0.05810854  0.04774289 -0.02614286 -0.004891579  0.003255150  0.007419088  0.002147445 -0.001023454
2 -2.965367  0.11149886 -0.09160923  0.05016301  0.009385979 -0.006245992 -0.014235772 -0.004120525  0.001963807
        Within cluster sum of squares by cluster:                    0   1
        [1]  601.325 2398.223                                    1 434  15
         (between_SS / total_SS =  51.1 %)                       2  10 224
```

The inaccuracy was same as the original k-means clustering, 3.66%. Clusters are the same because both the original K-means algorithm and PCA weigh the most important factors or combinations more.

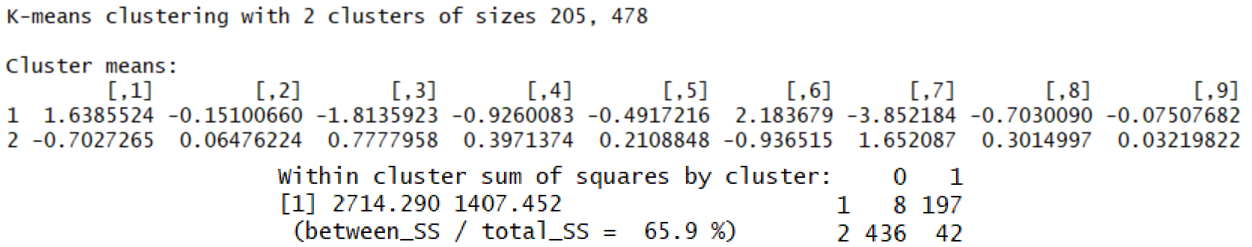
2.2.3.2 K-Means Clustering on ICA



```
K-means clustering with 2 clusters of sizes 538, 145

Cluster means:
        [,1]       [,2]       [,3]       [,4]       [,5]        [,6]       [,7]        [,8]        [,9]
1  0.0435723  0.1258125 -0.1625062 -0.1584953  0.3454873  0.02465808 -0.1971938  0.03569001 -0.07772547
2 -0.1616683 -0.4668078  0.6029542  0.5880722 -1.2818769 -0.09148999  0.7316570 -0.13242226  0.28838828
                     Within cluster sum of squares by cluster:     0   1
                     [1] 3144.345 2406.046                       1 433 105
                      (between_SS / total_SS =    9.7 %)         2  11 134
```

The error rate increased to 16.98% primarily due to lack of independent components present in the data. Clusters are different from original since the data is newly projected by independent components, which barely exist in this case.

2.2.3.3 K-Means Clustering on RP



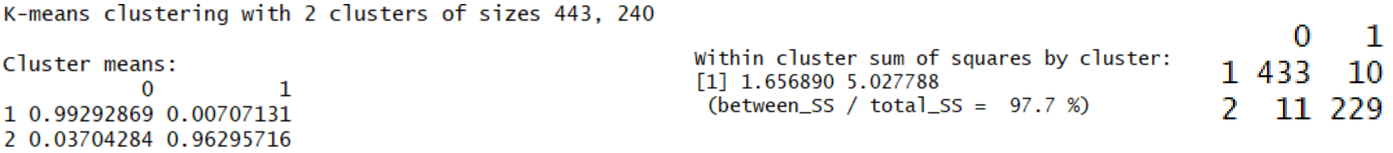
```
K-means clustering with 2 clusters of sizes 205, 478

Cluster means:
        [,1]        [,2]       [,3]       [,4]       [,5]      [,6]      [,7]       [,8]        [,9]
1  1.6385524 -0.15100660 -1.8135923 -0.9260083 -0.4917216  2.183679 -3.852184 -0.7030090 -0.07507682
2 -0.7027265  0.06476224  0.7777958  0.3971374  0.2108848 -0.936515  1.652087  0.3014997  0.03219822
```

```
Within cluster sum of squares by cluster:
[1] 2714.290 1407.452
 (between_SS / total_SS =  65.9 %)
```

```
    0   1
1   8 197
2 436  42
```

The error rate is higher than when k-means is performed on PCA but lower than on ICA, 7.32%, somewhat due to randomness. Clusters are different from original due to randomness as well.

2.2.3.4 K-Means Clustering on LDA



```
K-means clustering with 2 clusters of sizes 443, 240

Cluster means:
           0          1
1 0.99292869 0.00707131
2 0.03704284 0.96295716
```

```
Within cluster sum of squares by cluster:
[1] 1.656890 5.027788
 (between_SS / total_SS =  97.7 %)
```

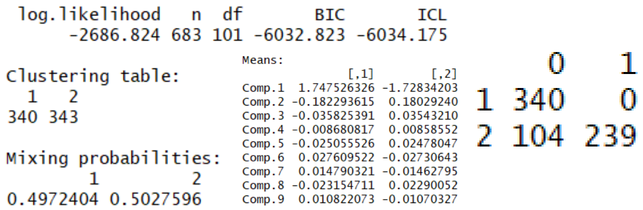
```
    0   1
1 433  10
2  11 229
```

The between sum of squares divided by total sum of squares is high, which means it predicts well enough.

It misclassified 3.07% of the data, which is the best result among when each K-Means clustering algorithm is performed on dimensionality reduction algorithm.

2.2.3.5 EM Clustering on PCA

Every statistics and the error rate, 15.23%, stay the same as the EM algorithm performed on the original data, but the coordinates for the clusters' centers are different.



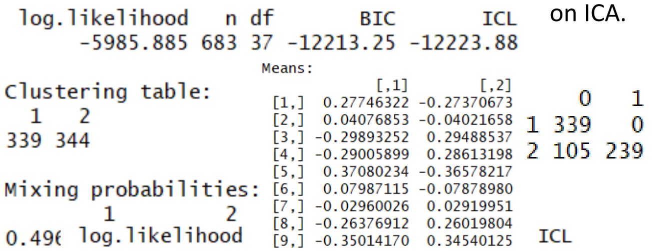
```
 log.likelihood   n  df       BIC       ICL
      -2686.824 683 101 -6032.823 -6034.175

Clustering table:
  1   2
340 343

Mixing probabilities:
        1         2
0.4972404 0.5027596
```

```
Means:
               [,1]        [,2]
Comp.1  1.747526326 -1.72834203
Comp.2 -0.182293615  0.18029240
Comp.3 -0.035825391  0.03543210
Comp.4 -0.008680817  0.00858552
Comp.5 -0.025055526  0.02478047
Comp.6  0.027609522 -0.02730643
Comp.7  0.014790321 -0.01462795
Comp.8 -0.023154711  0.02290052
Comp.9  0.010822073 -0.01070327
```

```
    0   1
1 340   0
2 104 239
```

2.2.3.6 EM Clustering on ICA

Here is the summary of results after running EM clustering algorithm on ICA.



```
 log.likelihood   n df       BIC      ICL
      -5985.885 683 37 -12213.25 -12223.88

Clustering table:
  1   2
339 344

Mixing probabilities:
        1         2
0.49(
```

```
Means:
             [,1]        [,2]
[1,]  0.27746322 -0.27370673
[2,]  0.04076853 -0.04021658
[3,] -0.29893252  0.29488537
[4,] -0.29005899  0.28613198
[5,]  0.37080234 -0.36578217
[6,]  0.07987115 -0.07878980
[7,] -0.02960026  0.02919951
[8,] -0.26376912  0.26019804
[9,] -0.35014170  0.34540125
```

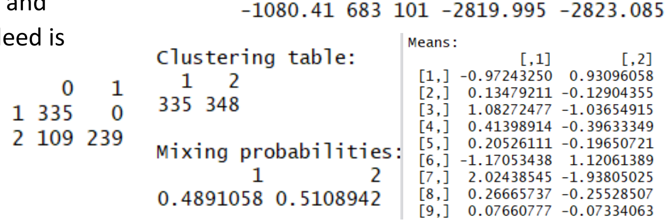
```
    0   1
1 339   0
2 105 239
```

Compared to the original EM algorithm, the accuracy slightly decreased along with the log likelihood and BIC, 15.37%.

2.2.3.7 EM Clustering on RP

According to the summary of this combination, the log likelihood and the BIC values are much higher than others, and the accuracy indeed is lower than others with 15.96% error rate.

This result is subject to change due to the inherent randomness in the RP algorithm. Clusters are different from original since the data is randomly projected.



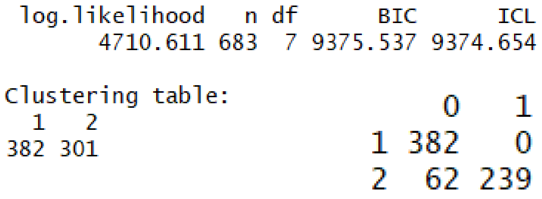
```
 log.likelihood                               ICL
       -1080.41 683 101 -2819.995 -2823.085

Clustering table:
  1   2
335 348

Mixing probabilities:
        1         2
0.4891058 0.5108942
```

```
Means:
             [,1]        [,2]
[1,] -0.97243250  0.93096058
[2,]  0.13479211 -0.12904355
[3,]  1.08272477 -1.03654915
[4,]  0.41398914 -0.39633349
[5,]  0.20526111 -0.19650721
[6,] -1.17053438  1.12061389
[7,]  2.02438545 -1.93805025
[8,]  0.26665737 -0.25528507
[9,]  0.07660777 -0.07334063
```

```
    0   1
1 335   0
2 109 239
```

2.2.3.8 EM Clustering on LDA

Differently than any other case in this section, this combination of methods results in under 10% error rate: 9.08%, with highest log likelihood value and BIC. Clusters are different from original since the data is reduced.



```
 log.likelihood   n df      BIC      ICL
       4710.611 683  7 9375.537 9374.654

Clustering table:
  1   2
382 301
```

```
    0   1
1 382   0
2  62 239
```

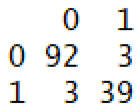
2.2.4 Neural Network on Dimensionality Reduction

In addition to the combinations of those algorithms like above, artificial neural network algorithm is added to see if any of the dimensionality algorithm improves the prediction accuracy. Then, the best model is compared to the recommended model in the Assignment 1. For more details about the artificial neural network, refer to the analysis in Assignment 1 or go to http://neuralnetworksanddeeplearning.com/.

In this section and the section 2.2.5, the breast cancer dataset is divided into training and testing sets with ratio of 8:2, and visualization step is skipped since it is not very intuitive. All the ANN algorithms are performed with default parameters, such as weights, and so on, except the number of hidden layers as 3, which is calculated by rounding the squared root value of the different between the number of inputs and the number of outputs. These models can be further improved using customized optimal set of parameters. The model is fit by using the 'nnet' method in R.

2.2.4.1 Neural Network on PCA

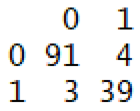
When the artificial neural network (ANN) algorithm is performed on the newly projected data by PCA, the error rate is 3.47%, which is so far the least. The model is a 9-3-1 network with 34 weights.



```
   0  1
0 92  3
1  3 39
```

2.2.4.2 Neural Network on ICA

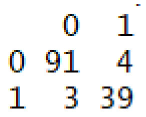
Also, the ANN algorithm on the data projected by ICA improved the prediction rate as the same as the one above, ANN on PCA, which results in 4.05% error rate. The model is also 9-3-1 network with 34 weights. This is significant because the performances ICA algorithm and its combination with other algorithms have been worse than any others.



```
   0  1
0 91  4
1  3 39
```

2.2.4.3 Neural Network on RP

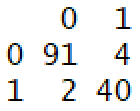
The ANN algorithm on randomly projected data results in 4.05% error rate probably due to randomness. It also results in a 9-3-1 network with 34 weights. This can be improved by running the RP algorithm multiple times and normalize.



```
   0  1
0 91  4
1  3 39
```

2.2.4.4 Neural Network on LDA

Since LDA algorithm reduces the dimension to 2, a 2-2-1 network with 9 weights is used, and this combination results in 2.92% error rate. When neural network algorithm is performed on the data projected by dimensionality reduction algorithm, PCA and ICA worked the best with 3.47% error rate.



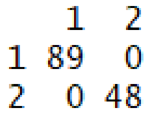
```
   0  1
0 91  4
1  2 40
```

2.2.5 Neural Network on Clustering on Dimensionality Reduction

In this section, the ANN algorithm is performed on clusters that are rendered on the dimensionality reduction algorithms introduced in this report to see whether the prediction rate increases or decreases.

2.2.5.1 Neural Network on K-Means on PCA

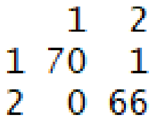
Surprisingly, the prediction rate results in 100%, after running the ANN algorithm on k-means clusters that are run on the newly projected data by the PCA algorithm. Also, a 9-3-1 network with 34 weights is used.



```
   1  2
1 89  0
2  0 48
```

2.2.5.2 Neural Network on EM on PCA

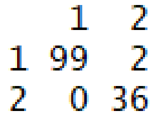
Also, when the ANN algorithm is performed on EM after PCA, the prediction rate is nearly 100% with a 9-3-1 network and 34 weights. Only one out of the 137 observations in the testing set is wrongfully predicted, 0.73% error, which is a significant decrease from 15.23% error rate when only EM is performed on PCA data.



```
   1  2
1 70  1
2  0 66
```

2.2.5.2 Neural Network on K-Means on ICA

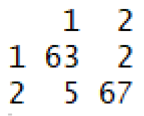
In this case, the accuracy falls slightly to where 2 predictions are missed. This is, however, also a significant increase from when only k-means algorithm is performed on ICA data, 16.98% to 1.45% error.



```
   1  2
1 99  2
2  0 36
```

2.2.5.2 Neural Network on EM on ICA

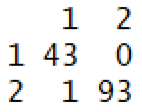
This combination results in a slight worse accuracy rate compared to the previous ones: 7 out of 137 are wrong, 5.11%. However, this is still a significant improvement from the combination of only EM and ICA, 15.37%.



```
   1  2
1 63  2
2  5 67
```

2.2.5.2 Neural Network on K-Means on RP

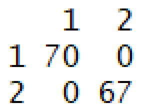
This combination of methods leads to the same output as in the section 2.2.5.2, where the ANN is performed on EM on PCA: 1 case is predicted wrong, 0.73% error. This also shows a superb improvement from 7.32% when only the EM is run on the RP data. This can be improved when repetitive runs fully normalize the RP data.



```
   1  2
1 43  0
2  1 93
```

2.2.5.2 Neural Network on EM on RP

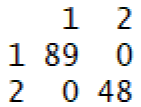
Again, with a much surprise, this way of combining the algorithms results in 100% accuracy when predicting whether the breast cancer is present or not. The improvement from the 15.96% error rate to 0% is indescribably significant. This result, however, can be only due to randomness and can be degraded if run with different random seed.



```
   1  2
1 70  0
2  0 67
```

2.2.5.2 Neural Network on K-Means on LDA

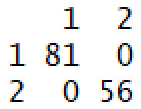
Again, this case results in 100% accuracy, which is also a substantial decrease from 3.07% when only K-Means algorithm is performed on LDA without ANN.



```
   1  2
1 89  0
2  0 48
```

2.2.5.2 Neural Network on EM on LDA

Yet again, 100% accuracy is achieved when ANN is performed on EM clusters made by LDA-projected data.



```
   1  2
1 81  0
2  0 56
```

2.3 Conclusion

The purpose of this problem was to correctly identify whether someone has a breast cancer given phycial and biological characteristics of various wine. The lowest error rate, 0%, could be achieved with many combinations of algorithms: when the neural network was used on the data clustered by k-means algorithm that is used on the newly projected data by the PCA, when the neural network was used on the data clustered by k-means algorithm that is used on the newly projected data by the LDA, and the neural network was used on the data clustered by EM algorithm that is used on the newly projected data by the LDA due to the problem domain and problem simplicity and the correctness and efficiency of neural network algorithm in general. Additionally, comparing to the neural network algorithm used in the Assignment 1, where the accuracy remains about 97%, the performances in those combinations introduced above end up being much better. The speed to calculate is negligibly different. Finally, here is the summary table of results with highlight on error rate less than 2%:

Problem 1	Error Rate	Problem 2	Error Rate		
K-Means	3.37%	K-Means	3.66%	NN on ICA	4.05 %
EM	2.25%	EM	15.23%	NN on RP	4.05%
KM on PCA	3.37%	KM on PCA	3.66%	NN on LDA	3.47%
KM on ICA	3.393%	KM on ICA	16.98%	NN on KM on PCA	0%
KM on RP	12.36%	KM on RP	7.32%	NN on EM on PCA	0.73%
KM on LDA	0%	KM on LDA	3.07%	NN on KM on ICA	1.45%
EM on PCA	2.81%	EM on PCA	15.23%	NN on EM on ICA	5.11%
EM on ICA	16.85%	EM on ICA	15.37%	NN on KM on RP	0.73%
EM on RP	1.69%	EM on RP	15.96%	NN on EM on RP	0%
EM on LDA	0.562%	EM on LDA	9.08%	NN on KM on LDA	0%
		NN on PCA	3.47%	NN on EM on LDA	0%

Overall, when NN algorithm is added to any combination, the result is satisfying due to its specific learnability.