Data Mining & Neural Networks Computational Task 2

Luke Emmins 189025797

Task 1

In the previous computational task, standardised and normalised data was used so there was no need to adjust the data because the same dataset was used. To find the principal components for the dataset, opted to use MATLAB’s built-in pca function. This gave me the coefficients for each term, so the data projection could happen further on in the assignment. To find the individual principal components, I found the eigenvalues of the covariance matrix of the data. The following table represents the attributes with their associated eigenvalue, loads and principal component number.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Principal Component | Eigenvalue | Load (%) |
| Radius | 30 | 0.000133 | 0.00044 |
| Area | 29 | 0.0007488 | 0.0025 |
| Perimeter | 28 | 0.0016 | 0.0053 |
| Texture | 27 | 0.0069 | 0.023 |
| Smoothness | 26 | 0.0082 | 0.027 |
| Compactness | 25 | 0.0155 | 0.052 |
| Concavity | 24 | 0.0181 | 0.060 |
| Mean Concave Points | 23 | 0.0243 | 0.081 |
| Symmetry | 22 | 0.0274 | 0.09 |
| Mean Fractal Dimension | 21 | 0.0300 | 0.10 |
| SE Radius | 20 | 0.0312 | 0.10 |
| SE Area | 19 | 0.0495 | 0.16 |
| SE Perimeter | 18 | 0.0526 | 0.18 |
| SE Texture | 17 | 0.0594 | 0.20 |
| SE Smoothness | 16 | 0.0799 | 0.27 |
| SE Compactness | 15 | 0.0941 | 0.31 |
| SE Concavity | 14 | 0.1570 | 0.52 |
| SE Mean Concave Points | 13 | 0.2414 | 0.80 |
| SE Symmetry | 12 | 0.2612 | 0.87 |
| SE Mean Fractal Dimension | 11 | 0.2939 | 0.98 |
| Worst Radius | 10 | 0.3507 | 1.17 |
| Worst Area | 9 | 0.4169 | 1.39 |
| Worst Perimeter | 8 | 0.4766 | 1.59 |
| Worst Texture | 7 | 0.6752 | 2.25 |
| Worst Smoothness | 6 | 1.2074 | 4.02 |
| Worst Compactness | 5 | 1.6487 | 5.50 |
| Worst Concavity | 4 | 1.9806 | 6.60 |
| Worst Mean Concave Points | 3 | 2.8179 | 9.39 |
| Worst Symmetry | 2 | 5.6914 | 18.97 |
| Worst Mean Fractal Dimension | 1 | 13.2816 | 44.27 |

In the table, the principal component value is dependent on its eigenvalue (and eigenvector), and the load represents the total amount of variance explained by its principal component. Using this, I can produce a plot that shows the eigenvalue so we can look closer at what number of components Chart, line chart

Description automatically generatedcan be retained.

The Kaiser rule states that for an eigenvalue of 1, as a principal component, then a factor contains the same amount of information as a single variable[[1]](#footnote-1). As a result of this, we only retain the principal components that are above this value of 1. As you can see from the graph, we retain only the first 6 principal components. If the graph is not clear enough, you can look at the eigenvalues in the table. Furthermore, we can also look at the conditional number rule, that states that such that the major components up to should be discarded. A major component was one that was retained from the Kaiser Rule. Therefore, the first 4 principal components are retained. Using these principal components, we can attempt to project the data into a single dimension. When we use this process, we sacrifice some accuracy for simplicity.

Task 2

Chart, histogram

Description automatically generatedThe first part of this task was to use the coefficients found from the MATLAB function to project the data down. Using the first three components, I can produce three separate histograms to find their separation to hopefully classify them with sufficiently high accuracy. As you would expect, the eigenvalues that explain the most variance would show a better split between data because there is more variance. The histograms below show this.

Chart, histogram

Description automatically generatedChart, histogram

Description automatically generated

So, as you can see, the first principal component shows a much better separation in comparison to the second and third. The below tables show the accuracy for each.

|  |  |  |  |
| --- | --- | --- | --- |
| Principal Component | Optimal Cut | Number of Misclassified | Accuracy |
| Pc1 | 0.0826 | 47 | 91.74% |
| Pc2 | 0.3726 | 212 | 62.74% |
| Pc3 | 0.3673 | 208 | 63.27% |

These predictors now need to be compared to the one attribute predictors used in computational task 1, the table below is a reminder of these values.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attribute | Optimal cut value | Malignant Accuracy | Benign Accuracy | Accuracy |
| Radius | 0.1107 | 81.13% | 90.48% | 88.93% |
| Area | 0.3673 | 55.19% | 81.51% | 63.27% |
| Perimeter | 0.2671 | 75.47% | 97.2% | 73.29% |
| Texture | 0.1090 | 76.42% | 96.92% | 89.1% |
| Smoothness | 0.1072 | 63.68% | 67.23% | 89.28% |
| Compactness | 0.3234 | 64.15% | 89.36% | 67.66% |
| Concavity | 0.1968 | 77.36% | 94.12% | 80.32% |
| Mean Concave Points | 0.1142 | 87.26% | 93.0% | 88.58% |
| Symmetry | 0.0861 | 59.43% | 67.23% | 91.39% |
| Mean Fractal Dimension | 0.3111 | 33.02% | 71.15% | 68.89% |

As I only used the first 10 attributes in task 1, I can only compare to these. However, as I previously stated, when projecting the data down you lose some accuracy. This is interesting because the accuracy of the classification is still higher than any found using one attribute predictors. This is because of the attribute with the highest principal component not being looked at as a single attribute predictor.

Finally, for task 2 I needed to plot the projected data onto different combinations of the principal axis.

Chart, scatter chart

Description automatically generatedChart, scatter chart

Description automatically generatedChart, scatter chart

Description automatically generated

Figure 2

Figure 1

Figure 3

From first glance, you can clearly see that the scatter graphs with the first principal component in shows a clearer separation between Malignant and Benign data. In figure 1, the two principal components that explain the highest variance represent the best class separation with two clear clusters of data and several points overlapping in the centre. Furthermore, as the components with less variance explained are used, more and more points start to overlap. This is seen in figure 3, when two components are used that explain minimal variance. There is no separation of classes here.

Task 3

In task 3 I needed to perform k-means clustering for , this was done using MATLAB’s built-in function. As the k-means clustering did not always produce the same centroids and clusters, we were required to produce multiple sets of results and then find the best one and repeat for each value of Below, you will find the tables for each value of along with their centroids and their Davies-Boulding index.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Indicator | Attempt 1 | | Attempt 2 | | Attempt 3 | | Attempt 4 | | Attempt 5 | |
|  |  |  |  |  |  |  |  |  |  |
| Radius | -0.49998 | 0.966454 | -0.48635 | 0.985631 | -0.48257 | 0.932803 | -0.484 | 0.97312 | 0.956129 | -0.49078 |
| Area | -0.47772 | 0.92343 | -0.24257 | 0.491589 | -0.23883 | 0.461658 | -0.23928 | 0.481091 | 0.470001 | -0.24125 |
| Perimeter | -0.31073 | 0.600638 | -0.50221 | 1.017771 | -0.49998 | 0.966454 | -0.50023 | 1.00575 | 0.98868 | -0.50749 |
| Texture | -0.52556 | 1.015895 | -0.48058 | 0.97394 | -0.47772 | 0.92343 | -0.47881 | 0.96268 | 0.943969 | -0.48454 |
| Smoothness | -0.57853 | 1.118286 | -0.28945 | 0.586601 | -0.31073 | 0.600638 | -0.30276 | 0.608718 | 0.590244 | -0.30297 |
| Compactness | -0.58665 | 1.133998 | -0.49994 | 1.013182 | -0.52556 | 1.015895 | -0.50722 | 1.019799 | 1.009164 | -0.518 |
| Concavity | -0.30881 | 0.596931 | -0.56445 | 1.143916 | -0.57853 | 1.118286 | -0.56622 | 1.138428 | 1.12072 | -0.57526 |
| Mean Concave Points | -0.14901 | 0.288034 | -0.57695 | 1.169254 | -0.58665 | 1.133998 | -0.57872 | 1.163558 | 1.140397 | -0.58536 |
| Symmetry | -0.42761 | 0.826564 | -0.29747 | 0.60286 | -0.30881 | 0.596931 | -0.30369 | 0.610601 | 0.59635 | -0.30611 |
| Mean Fractal Dimension | -0.02109 | 0.040769 | -0.11303 | 0.229073 | -0.14901 | 0.288034 | -0.12534 | 0.252008 | 0.248201 | -0.1274 |
| SE Radius | -0.43059 | 0.832331 | -0.42552 | 0.862358 | -0.42761 | 0.826564 | -0.42666 | 0.857842 | 0.830814 | -0.42646 |
| SE Area | -0.4014 | 0.775899 | -0.02177 | 0.044125 | -0.02109 | 0.040769 | -0.02124 | 0.042703 | 0.023665 | -0.01215 |
| SE Perimeter | -0.02381 | 0.046025 | -0.42619 | 0.863705 | -0.43059 | 0.832331 | -0.4275 | 0.859523 | 0.836306 | -0.42927 |
| SE Texture | -0.36988 | 0.714965 | -0.4012 | 0.813061 | -0.4014 | 0.775899 | -0.40108 | 0.806398 | 0.783445 | -0.40214 |
| SE Smoothness | -0.3343 | 0.646205 | -0.00606 | 0.012279 | -0.02381 | 0.046025 | -0.00848 | 0.017046 | 0.005759 | -0.00296 |
| SE Compactness | -0.39736 | 0.768098 | -0.34156 | 0.69221 | -0.36988 | 0.714965 | -0.34539 | 0.694439 | 0.690928 | -0.35465 |
| SE Concavity | -0.07486 | 0.144712 | -0.31541 | 0.639203 | -0.3343 | 0.646205 | -0.31649 | 0.636335 | 0.630681 | -0.32373 |
| SE Mean Concave Points | -0.23661 | 0.457373 | -0.38044 | 0.770989 | -0.39736 | 0.768098 | -0.38574 | 0.775556 | 0.75734 | -0.38874 |
| SE Symmetry | -0.5174 | 1.000126 | -0.06803 | 0.137866 | -0.07486 | 0.144712 | -0.06976 | 0.140259 | 0.125638 | -0.06449 |
| SE Mean Fractal Dimension | -0.25126 | 0.485677 | -0.1991 | 0.403495 | -0.23661 | 0.457373 | -0.20624 | 0.414667 | 0.414374 | -0.2127 |
| Worst Radius | -0.53184 | 1.028048 | -0.51875 | 1.051288 | -0.5174 | 1.000126 | -0.51685 | 1.039169 | 1.021163 | -0.52416 |
| Worst Area | -0.49892 | 0.964405 | -0.25491 | 0.516602 | -0.25126 | 0.485677 | -0.2516 | 0.505865 | 0.492634 | -0.25287 |
| Worst Perimeter | -0.3158 | 0.610439 | -0.53131 | 1.076747 | -0.53184 | 1.028048 | -0.52971 | 1.065034 | 1.048489 | -0.53819 |
| Worst Texture | -0.49181 | 0.950661 | -0.49987 | 1.013026 | -0.49892 | 0.964405 | -0.4985 | 1.002272 | 0.982636 | -0.50439 |
| Worst Smoothness | -0.53637 | 1.0368 | -0.29484 | 0.597518 | -0.3158 | 0.610439 | -0.30228 | 0.607758 | 0.597256 | -0.30657 |
| Worst Compactness | -0.57959 | 1.120343 | -0.46976 | 0.952017 | -0.49181 | 0.950661 | -0.4725 | 0.950001 | 0.950962 | -0.48813 |
| Worst Concavity | -0.30309 | 0.585867 | -0.51837 | 1.050518 | -0.53637 | 1.0368 | -0.51894 | 1.04338 | 1.043694 | -0.53573 |
| Worst Mean Concave Points | -0.33643 | 0.650321 | -0.56858 | 1.152275 | -0.57959 | 1.120343 | -0.56959 | 1.145203 | 1.130997 | -0.58054 |
| Worst Symmetry | -0.49998 | 0.966454 | -0.29551 | 0.598886 | -0.30309 | 0.585867 | -0.29687 | 0.596891 | 0.591253 | -0.30349 |
| Worst Mean Fractal Dimension | -0.47772 | 0.92343 | -0.30252 | 0.613081 | -0.33643 | 0.650321 | -0.30932 | 0.621922 | 0.628903 | -0.32281 |
| DB Index | 0.6321 | | 0.6275 | | 0.6320 | | 0.6285 | | 0.6308 | |

Table 1 Representing the 2-means clustering values, where the entries represent the centroids.

Table 2 Representing the 3-means clustering values, where the entries represent the centroids.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Indicator | Attempt 1 | | | Attempt 2 | | | Attempt 2 | | | |
|  |  |  |  |  |  |  |  |  |
| Radius | -0.4633 | 1.554379 | -0.46333 | -0.21742 | 1.538961 | 1.538961 | -0.4477 | -0.17218 | 1.617659 |
| Area | -0.2483 | 0.600469 | -0.24837 | 0.165183 | 0.601914 | 0.601914 | -0.24164 | 0.178829 | 0.626062 |
| Perimeter | -0.4847 | 1.558622 | -0.48477 | -0.14371 | 1.543846 | 1.543846 | -0.46913 | -0.10066 | 1.622573 |
| Texture | -0.4619 | 1.586929 | -0.46195 | -0.26064 | 1.569107 | 1.569107 | -0.44792 | -0.21982 | 1.661691 |
| Smoothness | -0.3631 | 0.393287 | -0.36314 | 0.841943 | 0.404733 | 0.404733 | -0.35836 | 0.848479 | 0.398214 |
| Compactness | -0.5818 | 0.886367 | -0.58183 | 1.041095 | 0.892188 | 0.892188 | -0.57306 | 1.034521 | 0.929796 |
| Concavity | -0.6087 | 1.186999 | -0.60875 | 0.770169 | 1.186104 | 1.186104 | -0.59677 | 0.780319 | 1.238248 |
| Mean Concave Points | -0.6097 | 1.429781 | -0.60974 | 0.476411 | 1.421572 | 1.421572 | -0.59421 | 0.498988 | 1.485663 |
| Symmetry | -0.3469 | 0.420139 | -0.34695 | 0.76204 | 0.420212 | 0.420212 | -0.34181 | 0.752131 | 0.431791 |
| Mean Fractal Dimension | -0.2348 | -0.30493 | -0.23484 | 1.243856 | -0.29193 | -0.29193 | -0.24167 | 1.182941 | -0.28667 |
| SE Radius | -0.4287 | 1.35759 | -0.42874 | -0.10487 | 1.347806 | 1.347806 | -0.41757 | -0.08495 | 1.440018 |
| SE Area | -0.0324 | 0.064901 | -0.03244 | 0.041434 | 0.062887 | 0.062887 | -0.03523 | 0.023382 | 0.093738 |
| SE Perimeter | -0.4355 | 1.321006 | -0.43554 | -0.03167 | 1.309903 | 1.309903 | -0.42528 | -0.01781 | 1.404145 |
| SE Texture | -0.3976 | 1.339018 | -0.39766 | -0.19429 | 1.326905 | 1.326905 | -0.38782 | -0.17221 | 1.422253 |
| SE Smoothness | -0.0964 | -0.09176 | -0.09643 | 0.476796 | -0.09299 | -0.09299 | -0.10011 | 0.439265 | -0.07261 |
| SE Compactness | -0.4495 | 0.433582 | -0.44951 | 1.116177 | 0.442402 | 0.442402 | -0.44575 | 1.073742 | 0.478651 |
| SE Concavity | -0.3875 | 0.409722 | -0.38751 | 0.925131 | 0.410748 | 0.410748 | -0.38246 | 0.889725 | 0.439366 |
| SE Mean Concave Points | -0.4604 | 0.750493 | -0.46047 | 0.776719 | 0.743478 | 0.743478 | -0.45051 | 0.75148 | 0.787142 |
| SE Symmetry | -0.0902 | 0.021405 | -0.09024 | 0.31897 | 0.013694 | 0.013694 | -0.09638 | 0.286204 | 0.054379 |
| SE Mean Fractal Dimension | -0.3230 | 0.088684 | -0.32309 | 1.082089 | 0.096469 | 0.096469 | -0.3221 | 1.025515 | 0.118933 |
| Worst Radius | -0.5023 | 1.599351 | -0.50233 | -0.14082 | 1.593353 | 1.593353 | -0.4857 | -0.08815 | 1.665284 |
| Worst Area | -0.2603 | 0.52723 | -0.26032 | 0.294803 | 0.534544 | 0.534544 | -0.25289 | 0.311194 | 0.542422 |
| Worst Perimeter | -0.5213 | 1.588781 | -0.52131 | -0.05568 | 1.581937 | 1.581937 | -0.50527 | -0.00701 | 1.655385 |
| Worst Texture | -0.4868 | 1.606385 | -0.48685 | -0.2048 | 1.59833 | 1.59833 | -0.47228 | -0.15634 | 1.683466 |
| Worst Smoothness | -0.3776 | 0.345304 | -0.37768 | 0.949943 | 0.362127 | 0.362127 | -0.37286 | 0.968298 | 0.336592 |
| Worst Compactness | -0.54024 | 0.638426 | -0.54024 | 1.178488 | 0.660739 | 0.660739 | -0.53354 | 1.178735 | 0.669702 |
| Worst Concavity | -0.5730 | 0.858033 | -0.57303 | 1.035851 | 0.870396 | 0.870396 | -0.56227 | 1.044136 | 0.885832 |
| Worst Mean Concave Points | -0.6124 | 1.240608 | -0.61249 | 0.713698 | 1.241837 | 1.241837 | -0.59655 | 0.742345 | 1.272063 |
| Worst Symmetry | -0.3186 | 0.302531 | -0.3186 | 0.799539 | 0.306897 | 0.306897 | -0.31735 | 0.809204 | 0.300082 |
| Worst Mean Fractal Dimension | -0.4127 | 0.069267 | -0.41277 | 1.412489 | 0.099445 | 0.099445 | -0.41108 | 1.383396 | 0.083992 |
| DB Index | 0.7276 | | | 0.7277 | | | 0.7231 | | |

Table 3 Representing the 5-means DBI values

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Indicator | Attempt 1 | Attempt 2 | Attempt 3 | Attempt 4 | Attempt 5 |
| DB Index | 0.9028 | 0.8869 | 0.9006 | 0.8836 | 0.8909 |

For 2-means clustering, there was a repetition of the three values in the table, so I stopped after those 3 attempts. Also, for 5-means clustering, the table of data for the centroids would be sufficiently large, so below I have included the centroids for the attempt that had the best clustering. The DB index will be explained further.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute |  |  |  |  |  |
| Radius | -0.28041 | 1.585839 | -0.90981 | -0.00423 | 0.863791 |
| Area | -0.24417 | 0.613932 | -0.26333 | 0.260678 | 0.39332 |
| Perimeter | -0.31705 | 1.570686 | -0.89202 | 0.066469 | 1.035117 |
| Texture | -0.32135 | 1.632138 | -0.80808 | -0.08284 | 0.850049 |
| Smoothness | -0.64616 | 0.267676 | 0.316693 | 0.85197 | 1.500772 |
| Compactness | -0.68606 | 0.671645 | -0.26256 | 1.050983 | 2.837932 |
| Concavity | -0.64093 | 1.049651 | -0.50977 | 0.828017 | 2.683741 |
| Mean Concave Points | -0.61166 | 1.341773 | -0.58586 | 0.609593 | 2.003157 |
| Symmetry | -0.55748 | 0.192126 | 0.171547 | 0.706272 | 2.531702 |
| Mean Fractal Dimension | -0.5364 | -0.48408 | 0.593304 | 0.964507 | 2.080874 |
| SE Radius | -0.47165 | 1.3192 | -0.3249 | -0.07623 | 1.6245 |
| SE Area | -0.25181 | -0.01452 | 0.455439 | -0.07393 | 1.048059 |
| SE Perimeter | -0.48351 | 1.232621 | -0.31358 | 0.006754 | 1.865552 |
| SE Texture | -0.39032 | 1.325519 | -0.42078 | -0.12865 | 1.295661 |
| SE Smoothness | -0.47127 | -0.14621 | 0.905314 | 0.170126 | 0.639204 |
| SE Compactness | -0.58275 | 0.268127 | 0.034712 | 0.821795 | 2.859505 |
| SE Concavity | -0.46799 | 0.308192 | -0.0908 | 0.60874 | 2.880853 |
| SE Mean Concave Points | -0.57473 | 0.658109 | -0.05334 | 0.553601 | 2.224078 |
| SE Symmetry | -0.30826 | -0.15346 | 0.434154 | 0.039408 | 2.475052 |
| SE Mean Fractal Dimension | -0.50142 | -0.00718 | 0.278108 | 0.714663 | 2.098141 |
| Worst Radius | -0.35033 | 1.629409 | -0.87727 | 0.087957 | 0.93853 |
| Worst Area | -0.24637 | 0.534204 | -0.32045 | 0.44937 | 0.378893 |
| Worst Perimeter | -0.3835 | 1.588345 | -0.85942 | 0.172758 | 1.153071 |
| Worst Texture | -0.3776 | 1.653379 | -0.76044 | -0.01407 | 0.866106 |
| Worst Smoothness | -0.61197 | 0.265347 | 0.199472 | 1.019883 | 0.943603 |
| Worst Compactness | -0.57143 | 0.490261 | -0.41279 | 1.258346 | 2.238398 |
| Worst Concavity | -0.57433 | 0.756909 | -0.54402 | 1.12618 | 2.218653 |
| Worst Mean Concave Points | -0.58061 | 1.165473 | -0.65731 | 0.873903 | 1.800947 |
| Worst Symmetry | -0.37655 | 0.125771 | -0.20079 | 0.840247 | 2.216099 |
| Worst Mean Fractal Dimension | -0.55829 | -0.0421 | 0.019875 | 1.356073 | 1.73348 |
| DB Index |  | | | | |

The DB index is known as the Davies-Boulding Index and is a measure for the quality of a clustering. It is found that the lower this value is, the better the clustering is for the data. For the 2-means clustering, attempt 2 with DBI of 0.6275 was the best clustering, attempt 3 for the 3-means clustering with DBI of 0.7231 and for 5-means attempt 2 was best with DBI of 0.8836 was the best. Comparing these values, suggests that the 2-means clustering produced better clusters with regards to the features of the dataset[[2]](#footnote-2).

Now that we have the clusters, in smaller datasets, we can project them down into a lower dimension so that we can graphically see the plots for them. Instead of using 3 different principal components, I will use the first two as they represent the most variance in the original dataset. The following graphs represent the clusters along with the actual data classification, Malignant or Benign. The x-axis represents the first principal component and the y-axis represents the second principal component.

Chart, scatter chart

Description automatically generatedChart, scatter chart

Description automatically generatedChart, scatter chart

Description automatically generated

Figure 5

Figure 4

Figure 6

Starting with figure 4, as you can see, each cluster has majority of one class type with few points being misclassified. However, in figure 5, we see many more points being misclassified in some areas of the scatter graph, especially in the first cluster shown in red. The number of misclassified is all dependent on where the centroids are placed for each cluster. Also, one must remember that the cluster data has been projected down, so some accuracy has been sacrificed during the projection, which can explain only some of the misclassification. Finally, in figure 6, we see a good separation of classes in similar areas to the ones seen in the other figures. Again, it is the area in black that shows the most misclassified points, but the other areas show promising results. Furthermore, the area shown by the blue points show no misclassification even after the data projection, this is quite promising because in reality we want as little error in classification as possible.

Task 4

In the final task, I needed to get some information behind the scatter graphs that were plotted. Observations made do not reflect the classification accurately enough, this was why the purity of each cluster must be found. The purity was calculated by identifying the number of correctly classified points in each cluster and divide by the total amount of data points. The purity was found for each k-means by the following:

(for 2-means cluster, attempt 2)

|  |  |  |  |
| --- | --- | --- | --- |
| Cluster | Majority | Number of cases | Total |
| 1 | Benign | 344 | 381 |
| 2 | Malignant | 175 | 188 |
| total |  | 519 | 569 |

(for 3-means cluster)

|  |  |  |  |
| --- | --- | --- | --- |
| Cluster | Majority | Number of cases | Total |
| 1 | Malignant | 64 | 100 |
| 2 | Benign | 321 | 359 |
| 3 | Malignant | 110 | 110 |
| total |  | 495 | 569 |

(for 5-means cluster)

|  |  |  |  |
| --- | --- | --- | --- |
| Cluster | Majority | Number of cases | Total |
| 1 | Malignant | 30 | 47 |
| 2 | Benign | 196 | 208 |
| 3 | Malignant | 65 | 65 |
| 4 | Benign | 128 | 129 |
| 5 | Malignant | 104 | 120 |
| total |  | 523 | 569 |

If the purity of a clustering is close to 1, then we can say that the cluster is good enough to be used for classification. This is because the purity in essence is the number of correctly classified within each cluster out of the total, so if this is close to 1 all the data points were correctly classified. For the 2-means and 5-means, we found the purity to be over 90% which is a sufficiently high value. However, in my opinion this is still not high enough. This is because 1 in 10 people would be misdiagnosed if this classification was used and in reality, this is just not good enough.

Now, we can categorise each attribute and then find out the information gain it has. This is the last thing that needs to be done. We have already categorised each cluster for each k-means, so all we need to do is find out the information gain. This can be found below:

(for 2-means)

|  |  |  |  |
| --- | --- | --- | --- |
| Cluster | Majority | Number of cases | Total |
| 1 | Benign | 344 | 381 |
| 2 | Malignant | 175 | 188 |
| total |  | 519 | 569 |

(for 3-means)

|  |  |  |  |
| --- | --- | --- | --- |
| Cluster | Majority | Number of cases | Total |
| 1 | Malignant | 64 | 100 |
| 2 | Benign | 321 | 359 |
| 3 | Malignant | 110 | 110 |
| total |  | 495 | 569 |

(for 5-means)

|  |  |  |  |
| --- | --- | --- | --- |
| Cluster | Majority | Number of cases | Total |
| 1 | Malignant | 30 | 47 |
| 2 | Benign | 196 | 208 |
| 3 | Malignant | 65 | 65 |
| 4 | Benign | 128 | 129 |
| 5 | Malignant | 104 | 120 |
| total |  | 523 | 569 |

0.32862

|  |  |
| --- | --- |
| k-means | Relative Information Gain |
| 2 | 0.4795 |
| 3 | 0.5033 |
| 5 | 0.6550 |

What does the information gain tell us? For this case, the information gain tells us how helpful each k-means clustering was to the classification of the dataset, specifically how much ‘information’ about each class. [[3]](#footnote-3) Therefore, we can see that as the value of k for the k-means increased, we got better classification because we had more information stored for each class.[[4]](#footnote-4)

**References:**

Displayr. 2017. *Kaiser Rule*. [online] Available at: <https://docs.displayr.com/wiki/Kaiser\_Rule#:~:text=The%20more%20variables%20that%20load,is%20in%20summarizing%20the%20data.&text=An%20eigenvalue%20of%201.0%20means,information%20as%20a%20single%20variable.> [Accessed 29 March 2021].

Griffin, A., 2019. *PCA in matlab*. [online] Youtube.com. Available at: <https://www.youtube.com/watch?v=SLTo7z7Czjc> [Accessed 29 March 2021].

En.wikipedia.org. 2021. *Davies–Bouldin index - Wikipedia*. [online] Available at: <https://en.wikipedia.org/wiki/Davies%E2%80%93Bouldin\_index> [Accessed 29 March 2021].

En.wikipedia.org. 2021. *Davies–Bouldin index - Wikipedia*. [online] Available at: <https://en.wikipedia.org/wiki/Davies%E2%80%93Bouldin\_index> [Accessed 29 March 2021].

Sujan, N., 2021. *What is Entropy and why Information gain matter in Decision Trees?*. [online] Medium. Available at: <https://medium.com/coinmonks/what-is-entropy-and-why-information-gain-is-matter-4e85d46d2f01> [Accessed 29 March 2021].

**MATLAB Code (1)**

clear all

load('FinalDatabases.mat');

format long

R = corrcoef(CentralDatabase(:,2:31));

[V,eigenvalues] = eig(R);

eigenvalues = diag(eigenvalues);

[coeff,newdata,latend,tsd,variance] = pca(CentralDatabase(:,2:31));

figure(1)

plot([1:30],variance)

grid on

figure(2)

plot([1:30],eigenvalues(30:-1:1))

grid on

hold on

yline(1)

%first 3 component analysis

components = eigenvalues(28:30);

**MATLAB Code (2)**

coeff = pca(CentralDatabase(:,2:31));

label = CentralDatabase(:,1);

sum = 0;

for j = 1:30

for i = 1:30

p = coeff(j,i)^2;

sum = sum+p;

end

check(j) = sum;

sum = 0;

end

%data transformation for pc1

pc1 = CentralDatabase(:,2:31)\*coeff(:,1);

%split data into B & M

pc1 = [label pc1];

BenignTransformedpc1 = [];

MalignantTransformedpc1 = [];

for i = 1:569

if pc1(i,1) == 2

BenignTransformedpc1 = [BenignTransformedpc1; pc1(i,2)];

else

MalignantTransformedpc1 = [MalignantTransformedpc1; pc1(i,2)];

end

end

figure(1)

[N, edges] = histcounts(BenignTransformedpc1,50,'Normalization','probability');

for i = 1:length(edges)

Count = 0;

for j = 1:length(BenignTransformedpc1)

if BenignTransformedpc1(j) > edges(i)

Count = Count + 1;

else

Count = Count + 0;

end

end

for k = 1:length(MalignantTransformedpc1)

if MalignantTransformedpc1(k) <= edges(i)

Count = Count + 1;

else

Count = Count + 0;

end

end

Misclassification(i) = Count/569;

end

a = min(Misclassification);

figure(1)

histogram(MalignantTransformedpc1,20,'Normalization','probability')

hold on

histogram(BenignTransformedpc1,20,'Normalization','probability')

legend('Malignant','Benign')

xline(a, 'LineWidth',2)

%data transformation for pc2

pc2 = CentralDatabase(:,2:31)\*coeff(:,2);

%split data into B & M

pc2 = [label pc2];

BenignTransformedpc2 = [];

MalignantTransformedpc2 = [];

for i = 1:569

if pc2(i,1) == 2

BenignTransformedpc2 = [BenignTransformedpc2; pc2(i,2)];

else

MalignantTransformedpc2 = [MalignantTransformedpc2; pc2(i,2)];

end

end

figure(1)

[N, edges] = histcounts(BenignTransformedpc2,50,'Normalization','probability');

for i = 1:length(edges)

Count = 0;

for j = 1:length(BenignTransformedpc2)

if BenignTransformedpc2(j) > edges(i)

Count = Count + 1;

else

Count = Count + 0;

end

end

for k = 1:length(MalignantTransformedpc2)

if MalignantTransformedpc2(k) <= edges(i)

Count = Count + 1;

else

Count = Count + 0;

end

end

Misclassification(i) = Count/569;

end

b = min(Misclassification);

figure(2)

histogram(MalignantTransformedpc2,20,'Normalization','probability')

hold on

histogram(BenignTransformedpc2,20,'Normalization','probability')

legend('Malignant','Benign')

xline(b, 'LineWidth',2)

%data transformation for pc3

pc3 = CentralDatabase(:,2:31)\*coeff(:,3);

%split data into B & M

pc3 = [label pc3];

BenignTransformedpc3 = [];

MalignantTransformedpc3 = [];

for i = 1:569

if pc3(i,1) == 2

BenignTransformedpc3 = [BenignTransformedpc3; pc3(i,2)];

else

MalignantTransformedpc3 = [MalignantTransformedpc3; pc3(i,2)];

end

end

figure(1)

[N, edges] = histcounts(BenignTransformedpc3,50,'Normalization','probability');

for i = 1:length(edges)

Count = 0;

for j = 1:length(BenignTransformedpc3)

if BenignTransformedpc3(j) > edges(i)

Count = Count + 1;

else

Count = Count + 0;

end

end

for k = 1:length(MalignantTransformedpc3)

if MalignantTransformedpc3(k) <= edges(i)

Count = Count + 1;

else

Count = Count + 0;

end

end

Misclassification(i) = Count/569;

end

c = min(Misclassification);

figure(3)

histogram(MalignantTransformedpc3,20,'Normalization','probability')

hold on

histogram(BenignTransformedpc3,20,'Normalization','probability')

legend('Malignant','Benign')

xline(c, 'LineWidth',2)

%dataset plane comparison

%first & second

figure(4)

scatter(MalignantTransformedpc1,MalignantTransformedpc2,4)

hold on

scatter(BenignTransformedpc1,BenignTransformedpc2,4)

legend('Malignant','Benign')

%first & third

figure(5)

scatter(MalignantTransformedpc1,MalignantTransformedpc3,4)

hold on

scatter(BenignTransformedpc1,BenignTransformedpc3,4)

legend('Malignant','Benign')

%second and third

figure(6)

scatter(MalignantTransformedpc2,MalignantTransformedpc3,4)

hold on

scatter(BenignTransformedpc2,BenignTransformedpc3,4)

legend('Malignant','Benign')

**MATLAB Code (3)**

clear all

load('FinalDatabases.mat')

X = CentralDatabase(:,2:31);

[idx,C] = kmeans(X,2);

labels = CentralDatabase(:,1);

A = C;

%the vector of values in each cluster

clusterData = [idx X labels];

cluster1 = clusterData(clusterData(:,1)==1,:);

cluster2 = clusterData(clusterData(:,1)==2,:);

T = [length(cluster1(:,1)) length(cluster2(:,1))];

Sum = 0;

for i = 1

for j=1:T(i)

q = sqrt(sum((cluster1(j,2:31)-A(1,:)).^2));

Sum = Sum + q;

end

S(i) = (1/T(i) \* Sum)^(1/2);

end

Sum = 0;

for i = 2

for j=1:T(i)

q = sqrt(sum((cluster2(j,2:31)-A(2,:)).^2));

Sum = Sum + q;

end

S(i) = (1/T(i) \* Sum)^(1/2);

end

for i=1:2

for j=1:2

M(i,j) = norm(A(i,:)-A(j,:),2);

end

end

%assign diagonals large so calculatations can be made

M = M + diag([10000;100000]);

for i=1:2

for j=1:2

R(i,j) = (S(i)+S(j))/M(i,j);

end

end

for i=1:2

D(i) = max(R(i,:));

end

DB = 1/2 \* sum(D)

%need to project the data in each cluster using the principal components

coeff = pca(CentralDatabase(:,2:31));

pc1c1 = cluster1(:,2:31)\*coeff(:,1);

pc1c2 = cluster2(:,2:31)\*coeff(:,1);

pc2c1 = cluster1(:,2:31)\*coeff(:,2);

pc2c2 = cluster2(:,2:31)\*coeff(:,2);

figure(1)

scatter(pc1c1,pc2c1,'r.')

hold on

grid on

scatter(pc1c2,pc2c2)

%now to split clusters into M & B

%Malignant = 1 Benign = 2

cluster1M = cluster1(cluster1(:,32)==1,:);

cluster1B = cluster1(cluster1(:,32)==2,:);

cluster2M = cluster2(cluster2(:,32)==1,:);

cluster2B = cluster2(cluster2(:,32)==2,:);

%project each data into each principle axis

coeff = pca(CentralDatabase(:,2:31));

pc1Mc1 = cluster1M(:,2:31)\*coeff(:,1);

pc1Bc1 = cluster1B(:,2:31)\*coeff(:,1);

pc2Mc1 = cluster1M(:,2:31)\*coeff(:,2);

pc2Bc1 = cluster1B(:,2:31)\*coeff(:,2);

pc1Mc2 = cluster2M(:,2:31)\*coeff(:,1);

pc1Bc2 = cluster2B(:,2:31)\*coeff(:,1);

pc2Mc2 = cluster2M(:,2:31)\*coeff(:,2);

pc2Bc2 = cluster2B(:,2:31)\*coeff(:,2);

figure(2)

%malignant are circles, benign are triangles o ^

scatter(pc1Mc1,pc2Mc1,'o','.r')

hold on

grid on

scatter(pc1Bc1,pc2Bc1,'+','.r')

scatter(pc1Mc2,pc2Mc2,'o','g.')

scatter(pc1Bc2,pc2Bc2,'+','g.')

%purity calculation

%for cluster1 the majority is Benign

correctC1 = length(cluster1B(:,1));

%for cluster2 majority is Malignant

correctC2 = length(cluster2M(:,1));

Purity = (correctC1 + correctC2)/569

**MATLAB Code (4)**

clear all

load('FinalDatabases.mat')

X = CentralDatabase(:,2:31);

[idx,C] = kmeans(X,3);

labels = CentralDatabase(:,1);

A = C;

%the vector of values in each cluster

clusterData = [idx X labels];

cluster1 = clusterData(clusterData(:,1)==1,:);

cluster2 = clusterData(clusterData(:,1)==2,:);

cluster3 = clusterData(clusterData(:,1)==3,:);

T = [length(cluster1(:,1)) length(cluster2(:,1)) length(cluster3(:,1))];

Sum = 0;

for i = 1

for j=1:T(i)

q = sqrt(sum((cluster1(j,2:31)-A(1,:)).^2));

Sum = Sum + q;

end

S(i) = (1/T(i) \* Sum)^(1/2);

end

Sum = 0;

for i = 2

for j=1:T(i)

q = sqrt(sum((cluster2(j,2:31)-A(2,:)).^2));

Sum = Sum + q;

end

S(i) = (1/T(i) \* Sum)^(1/2);

end

Sum = 0;

for i = 3

for j=1:T(i)

q = sqrt(sum((cluster3(j,2:31)-A(3,:)).^2));

Sum = Sum + q;

end

S(i) = (1/T(i) \* Sum)^(1/2);

end

for i=1:3

for j=1:3

M(i,j) = norm(A(i,:)-A(j,:),2);

end

end

%assign diagonals large so calculatations can be made

M = M + diag([10000;100000;10000]);

for i=1:3

for j=1:3

R(i,j) = (S(i)+S(j))/M(i,j);

end

end

for i=1:3

D(i) = max(R(i,:));

end

DB = 1/3 \* sum(D)

%need to project the data in each cluster using the principal components

coeff = pca(CentralDatabase(:,2:31));

pc1c1 = cluster1(:,2:31)\*coeff(:,1);

pc1c2 = cluster2(:,2:31)\*coeff(:,1);

pc1c3 = cluster3(:,2:31)\*coeff(:,1);

pc2c1 = cluster1(:,2:31)\*coeff(:,2);

pc2c2 = cluster2(:,2:31)\*coeff(:,2);

pc2c3 = cluster3(:,2:31)\*coeff(:,2);

figure(1)

scatter(pc1c1,pc2c1,'r.')

hold on

grid on

scatter(pc1c2,pc2c2,'b.')

scatter(pc1c3,pc2c3,'g.')

%now to split clusters into M & B

%Malignant = 1 Benign = 2

cluster1M = cluster1(cluster1(:,32)==1,:);

cluster1B = cluster1(cluster1(:,32)==2,:);

cluster2M = cluster2(cluster2(:,32)==1,:);

cluster2B = cluster2(cluster2(:,32)==2,:);

cluster3M = cluster3(cluster3(:,32)==1,:);

cluster3B = cluster3(cluster3(:,32)==2,:);

%project each data into each principle axis

coeff = pca(CentralDatabase(:,2:31));

pc1Mc1 = cluster1M(:,2:31)\*coeff(:,1);

pc1Bc1 = cluster1B(:,2:31)\*coeff(:,1);

pc2Mc1 = cluster1M(:,2:31)\*coeff(:,2);

pc2Bc1 = cluster1B(:,2:31)\*coeff(:,2);

pc1Mc2 = cluster2M(:,2:31)\*coeff(:,1);

pc1Bc2 = cluster2B(:,2:31)\*coeff(:,1);

pc2Mc2 = cluster2M(:,2:31)\*coeff(:,2);

pc2Bc2 = cluster2B(:,2:31)\*coeff(:,2);

pc1Mc3 = cluster3M(:,2:31)\*coeff(:,1);

pc1Bc3 = cluster3B(:,2:31)\*coeff(:,1);

pc2Mc3 = cluster3M(:,2:31)\*coeff(:,2);

pc2Bc3 = cluster3B(:,2:31)\*coeff(:,2);

figure(2)

%malignant are circles, benign are triangles o ^

scatter(pc1Mc1,pc2Mc1,'o','.r')

hold on

grid on

scatter(pc1Bc1,pc2Bc1,'+','.r')

scatter(pc1Mc2,pc2Mc2,'o','g.')

scatter(pc1Bc2,pc2Bc2,'+','g.')

scatter(pc1Mc3,pc2Mc3,'o','y.')

scatter(pc1Bc3,pc2Bc3,'+','y.')

%purity calculation

%for cluster1 the majority is Malignant

correctC1 = length(cluster1M(:,1));

%for cluster2 majority is Benign

correctC2 = length(cluster2B(:,1));

%for cluster3 majority is Malignant

correctC3 = length(cluster3M(:,1));

Purity = (correctC1 + correctC2 + correctC3)/569

**MATLAB Code (5)**

clear all

load('FinalDatabases.mat')

X = CentralDatabase(:,2:31);

[idx,C] = kmeans(X,5);

labels = CentralDatabase(:,1);

A = C;

%the vector of values in each cluster

clusterData = [idx X labels];

cluster1 = clusterData(clusterData(:,1)==1,:);

cluster2 = clusterData(clusterData(:,1)==2,:);

cluster3 = clusterData(clusterData(:,1)==3,:);

cluster4 = clusterData(clusterData(:,1)==4,:);

cluster5 = clusterData(clusterData(:,1)==5,:);

T = [length(cluster1(:,1)) length(cluster2(:,1)) length(cluster3(:,1)) length(cluster4(:,1)) length(cluster5(:,1))];

Sum = 0;

for i = 1

for j=1:T(i)

q = sqrt(sum((cluster1(j,2:31)-A(1,:)).^2));

Sum = Sum + q;

end

S(i) = (1/T(i) \* Sum)^(1/2);

end

Sum = 0;

for i = 2

for j=1:T(i)

q = sqrt(sum((cluster2(j,2:31)-A(2,:)).^2));

Sum = Sum + q;

end

S(i) = (1/T(i) \* Sum)^(1/2);

end

Sum = 0;

for i = 3

for j=1:T(i)

q = sqrt(sum((cluster3(j,2:31)-A(3,:)).^2));

Sum = Sum + q;

end

S(i) = (1/T(i) \* Sum)^(1/2);

end

Sum = 0;

for i = 4

for j=1:T(i)

q = sqrt(sum((cluster4(j,2:31)-A(4,:)).^2));

Sum = Sum + q;

end

S(i) = (1/T(i) \* Sum)^(1/2);

end

Sum = 0;

for i = 5

for j=1:T(i)

q = sqrt(sum((cluster5(j,2:31)-A(5,:)).^2));

Sum = Sum + q;

end

S(i) = (1/T(i) \* Sum)^(1/2);

end

for i=1:5

for j=1:5

M(i,j) = norm(A(i,:)-A(j,:),2);

end

end

%assign diagonals large so calculatations can be made

M = M + diag([10000;100000;10000;100000;100000]);

for i=1:5

for j=1:5

R(i,j) = (S(i)+S(j))/M(i,j);

end

end

for i=1:5

D(i) = max(R(i,:));

end

DB = 1/5 \* sum(D)

%now to split clusters into M & B

%Malignant = 1 Benign = 2

cluster1M = cluster1(cluster1(:,32)==1,:);

cluster1B = cluster1(cluster1(:,32)==2,:);

cluster2M = cluster2(cluster2(:,32)==1,:);

cluster2B = cluster2(cluster2(:,32)==2,:);

cluster3M = cluster3(cluster3(:,32)==1,:);

cluster3B = cluster3(cluster3(:,32)==2,:);

cluster4M = cluster4(cluster4(:,32)==1,:);

cluster4B = cluster4(cluster4(:,32)==2,:);

cluster5M = cluster5(cluster5(:,32)==1,:);

cluster5B = cluster5(cluster5(:,32)==2,:);

%project each data into each principle axis

coeff = pca(CentralDatabase(:,2:31));

pc1Mc1 = cluster1M(:,2:31)\*coeff(:,1);

pc1Bc1 = cluster1B(:,2:31)\*coeff(:,1);

pc2Mc1 = cluster1M(:,2:31)\*coeff(:,2);

pc2Bc1 = cluster1B(:,2:31)\*coeff(:,2);

pc1Mc2 = cluster2M(:,2:31)\*coeff(:,1);

pc1Bc2 = cluster2B(:,2:31)\*coeff(:,1);

pc2Mc2 = cluster2M(:,2:31)\*coeff(:,2);

pc2Bc2 = cluster2B(:,2:31)\*coeff(:,2);

pc1Mc3 = cluster3M(:,2:31)\*coeff(:,1);

pc1Bc3 = cluster3B(:,2:31)\*coeff(:,1);

pc2Mc3 = cluster3M(:,2:31)\*coeff(:,2);

pc2Bc3 = cluster3B(:,2:31)\*coeff(:,2);

pc1Mc4 = cluster4M(:,2:31)\*coeff(:,1);

pc1Bc4 = cluster4B(:,2:31)\*coeff(:,1);

pc2Mc4 = cluster4M(:,2:31)\*coeff(:,2);

pc2Bc4 = cluster4B(:,2:31)\*coeff(:,2);

pc1Mc5 = cluster5M(:,2:31)\*coeff(:,1);

pc1Bc5 = cluster5B(:,2:31)\*coeff(:,1);

pc2Mc5 = cluster5M(:,2:31)\*coeff(:,2);

pc2Bc5 = cluster5B(:,2:31)\*coeff(:,2);

figure(1)

%malignant are circles, benign are triangles o ^

scatter(pc1Mc1,pc2Mc1,'o','.r')

hold on

grid on

scatter(pc1Bc1,pc2Bc1,'+','.r')

scatter(pc1Mc2,pc2Mc2,'o','g.')

scatter(pc1Bc2,pc2Bc2,'+','g.')

scatter(pc1Mc3,pc2Mc3,'o','y.')

scatter(pc1Bc3,pc2Bc3,'+','y.')

scatter(pc1Mc4,pc2Mc4,'o','b.')

scatter(pc1Bc4,pc2Bc4,'+','b.')

scatter(pc1Mc5,pc2Mc5,'o','k.')

scatter(pc1Bc5,pc2Bc5,'+','k.')

%need to project the data in each cluster using the principal components

pc1c1 = cluster1(:,2:31)\*coeff(:,1);

pc1c2 = cluster2(:,2:31)\*coeff(:,1);

pc1c3 = cluster3(:,2:31)\*coeff(:,1);

pc1c4 = cluster4(:,2:31)\*coeff(:,1);

pc1c5 = cluster5(:,2:31)\*coeff(:,1);

pc2c1 = cluster1(:,2:31)\*coeff(:,2);

pc2c2 = cluster2(:,2:31)\*coeff(:,2);

pc2c3 = cluster3(:,2:31)\*coeff(:,2);

pc2c4 = cluster4(:,2:31)\*coeff(:,2);

pc2c5 = cluster5(:,2:31)\*coeff(:,2);

%figure(2)

%scatter(pc1c1,pc2c1,'r.')

%hold on

%grid on

%scatter(pc1c2,pc2c2,'b.')

%scatter(pc1c3,pc2c3,'g.')

%scatter(pc1c4,pc2c4,'y.')

%scatter(pc1c5,pc2c5,'k.')

%purity calculation

%for cluster1 the majority is Benign

correctC1 = length(cluster1B(:,1));

%for cluster2 majority is Malignant

correctC2 = length(cluster2M(:,1));

%for cluster3 majority is Malignant

correctC3 = length(cluster3M(:,1));

%for cluster4 majority is Benign

correctC4 = length(cluster4B(:,1));

%for cluster5 majority is Malignant

correctC5 = length(cluster5M(:,1));

Purity = (correctC1 + correctC2 + correctC3 + correctC4 + correctC5)/569

1. Kaiser Rule - Displayr [↑](#footnote-ref-1)
2. Davies-Bouldin Index - Wikipedia [↑](#footnote-ref-2)
3. What is Entropy and why Information gain matter in Decision Trees? (Sujan, 2021) [↑](#footnote-ref-3)
4. Information Gain Ratio - Wikipedia [↑](#footnote-ref-4)