Using Unsupervised Learning to Investigate the Nature of Malignant and Benign Tumours

Tobias Meeks and Tom Roper

The paper showcases how unsupervised learning can be used to understand the features of Malignant and benign tumours within a breast cancer dataset. For our analyses, we apply three clustering algorithms (K-means, the Gaussian Mixture Model (GMM) and density based spatial clustering of applications with noise (DBSCAN)), as well as using principal component analysis (PCA). We select the best clustering algorithm by using two internal validation metrics (S_Dbw index and Silhouette scores) and use the best clusters, along with PCA, to learn about the features describing the clusters. Of interest, the Bare Nuclei feature captures the most variance in the dataset and displays the largest differences in average values between tumour types.

Introduction.

Unsupervised learning (UL) involves using algorithms to extract patterns and hidden structures within an unlabelled dataset. In this paper, we take this approach to study how the features of tumour samples vary based on whether they were malignant or benign. We consider the Wisconsin Breast Cancer Tumour dataset [1], with 699 data elements in total. Each entry corresponds to a single tumour sample from a patient and is characterised by 9 categorical features, with values from 0 to 10. After inspection, 16 entries had missing values in the 'Bare Nuclei' feature, which were completely removed. We felt justified in this decision as only a small number of entries needed to be removed, and it prevented the chance of unlikely data points being introduced into our analyses. The dataset also had two additional features; patient identification and tumour classification. We removed the identification label due to its irrelevance for understanding tumours, and we removed the classification label during modelling to replicate the standard UL paradigm (finding structures in an unlabelled dataset).

UL Modelling.

For use in our clustering algorithms and feature analysis, we transformed the dataset using PCA [2]. This is achieved by projecting the data onto a new coordinate system, described by the eigenvectors of the original data's covariance matrix. Firstly, PCA could condense a high amount (76%) of the original data's variance into its first two principal components, which would give us a more representative 2-D visualisation to assess the clustering of our data. Furthermore, by calculating correlation coefficients, we noticed that features describing Clump Size and Thickness were considerably correlated with each other (coefficient = 0.91) and most other features (coefficient ~ 0.7). We hypothesised that using the PCA-transformed data may also provide more optimal clustering results by decorrelating our features. We used the top 5 principal components (determined by largest eigenvalues) for our clustering algorithms, providing an optimal balance between 1) retaining most of the variance (91%) 2) higher clustering scores and 3) a meaningful reduction in DBSCAN noise values. The first two clustering techniques we applied were the GMM [3] and k-means clustering [4], selected for their simplicity and low computational cost. Both require a hyperparameter of K, that denotes the number of desired clusters. For K-means, in the first iteration, the algorithm initialises K number of centroids and creates clusters by assigning data points to their closest centroid (based on Euclidean distances). The initialised centroids are then replaced by the mean value of each cluster, and the process repeats until the cluster assignment is no longer changing between iterations. One disadvantage of K-means is that it assumes that all clusters are spherical in shape, which might not be true. The GMM provides an extension of K-means that allows for clusters to vary in shape to some degree. We take the centroids from K-means as the centres of gaussian distributions that describe each cluster. The likelihood that each data point belongs to any one cluster is calculated using a multivariate gaussian distribution:

$$N(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right) \tag{1}$$

Where d is the number of data dimensions, μ is the cluster's centroid, Σ is the cluster's covariance matrix and x is a data point. The GMM also works iteratively, optimising each cluster's: centroids, covariance matrices and mixing coefficients, until it has found gaussian distributions for each cluster that maximise the likelihood of observing the data (for more details [4]). Both K-means and GMM rely on an initial centroid choice; to ensure this choice was as optimal as possible we used the K++ initialisation method [5]. This selects the first centroid by picking a data point at random, and then picking further data points to be centroids with probabilities:

$$\text{probability} = \frac{D(x')^2}{\sum_{x\in X} D(x)^2}. \tag{2}$$

D(x) denotes the Euclidean distance between data point x and its closest (already picked) centroid and x' denotes a candidate for the next centroid. We also repeated both algorithms for 30 random initialisations of the first centroid and picked the best performing iteration (determined by our Silhouette score, see following section). We considered that GMM and K-means might be too simple to cluster our data effectively, so we also implemented the more complex algorithm DBSCAN [6]. DBSCAN creates clusters by connecting dense neighbourhoods of datapoints. It has hyper-parameters 'eps' which decides how large the radius of the datapoint's neighbourhood should be, and 'min points', the number of points needed within a neighbourhood for it to be considered sufficiently dense. Clusters are made by linking together contiguous regions of dense neighbourhoods. Without needing cluster centroids, DBSCAN removes the limitation of centroid initialisation and allows irregular cluster shapes to be made. Points that cannot be connected to dense neighbourhoods are considered noise points and are not assigned to a cluster. Importantly, K is a learnt parameter in DBSCAN, in contrast to being a hyperparameter in K-means and GMM.

Selecting the best clustering algorithm.

Next, we determine which algorithm has clustered the data most optimally. To provide more confidence in our validation measures we used two scoring metrics for this. First, we considered the S_Dbw index [7], as this measure is suggested to be robust, being able to validate clustering optimally across five common factors of dataset variability (such as being noisy or skewed) [8]. The S_Dbw score is the summation of the average inter-cluster density (Dens_bw) and average intra-cluster variance (Scat), given by:

$$Dens_{bw}(c) = \frac{1}{c(c-1)} \sum_{i=1}^{c} \sum_{j=1, j\neq i}^{c} \frac{density(u_{ij})}{\max(density(v_i), density(v_j))} \tag{3}$$

$$S_{cat}(c) = \frac{1}{c} \sum_{i=1}^{c} \frac{\|\sigma(v_i)\|}{\|\sigma(S)\|} \tag{4}$$

Where σ denotes the variance, c the number of clusters, v the centre of a cluster, S the entire dataset and u_{ij} the midpoint between cluster centres v_i and v_j. Data points contribute to the density of a centre (u or v) if the Euclidean distance between them is less than the average standard deviation of all clusters. Next, we looked at the Silhouette score [9], which is suggested to be nearly as robust as S_Dbw [8]. It calculates individual Silhouette scores for each data point as:

$$s_i = \frac{b_i - a_i}{\max(b_i, a_i)} \quad (5)$$

Where i is the ith datapoint, for which we calculate its average Euclidean distance to all other datapoints, either within (b_i) or not-within (a_i) its cluster. The final score is simply the average silhouette score across all datapoints. Given the excluded classification label, we expected our dataset to have a 2-dimensional hidden structure related to the Benign/Malignant tumour types, and subsequently focused our search in the vicinity of k=2, with a range from K=2-5. Stopping at 5 was decided to prevent overfitting and splitting our data into so many clusters that it would become difficult to generate meaningful and comparable insight. For an algorithm to be most optimal it should have the largest Silhouette score, and the smallest S_Dbw index. We used the top 5 PCs as input to the algorithms and, both before and after PCA, data had been scaled to equal ranges.

Clusters	Silhouette Score			S_Dbw		
	K-Means	GMM	DBSCAN	K-Means	GMM	DBSCAN
2	0.5880	0.4452	0.5996	0.6563	0.9972	0.6580
3	0.5421	0.5791	-	0.7361	0.9747	-
4	0.5388	0.5890	-	0.7872	1.0934	-
5	0.5443	0.6261	-	0.7827	0.8255	-

Table 1: Internal validation metrics for different clustering algorithms

Results are shown in table 1. DBSCAN (hyperparameters: ‘eps’ = 0.541 and ‘min points’ = 20) for k=2 clustered our data most optimally. Its silhouette score was the highest at 0.5996 (ignoring GMM) and was tied for the best S_Dbw index of 0.658 with K-means’ 0.6563. Despite only being marginally better than K-means for the Silhouette score, we also consider the benefit of DBSCAN removing noise values which improves the quality our final analysis. Across the board, (again, ignoring GMM) optimal scores were for k=2, matching the original classification label that segmented the dataset in 2 dimensions. Note the DBSCAN’s S_Dbw score may underpredict its performance, given that DBSCAN functions without a concept of centroids, but centroids are required and calculated for the S_Dbw index. Additionally, when calculating DBSCAN’s Silhouette score, we did not consider noise values as an independent cluster, but did include them in the inter-cluster distance calculation. Including them as a single cluster would wrongfully increase the average intra-cluster distance (worse score), but completely removing them would wrongfully increase the average inter-cluster distance (better score), our decision reflected a balance between these outcomes. For GMM, we found that our metrics were only accurate for k=2 (compared against official Python packages), which accounts for the fluctuating results for this model. We identified reasons for this inaccuracy in our score metrics, discussed further in our conclusion. Ultimately, we proceeded with DBSCAN for k=2 into the final section.

Understanding Benign and Malignant tumours.

To begin, we demonstrate the final clusters produced by DBSCAN. Normally, we might rely on domain expertise to identify what our clusters represent. In the absence of this we have used the original Benign/Malignant classifications [1] to help guide our cluster labelling. Of which, we have 458 tumour samples in cluster 1 (benign) and 194 in cluster 2 (malignant), with 47 points classified as noise. The results are visualised across PC1 and PC2 (Figure 1).

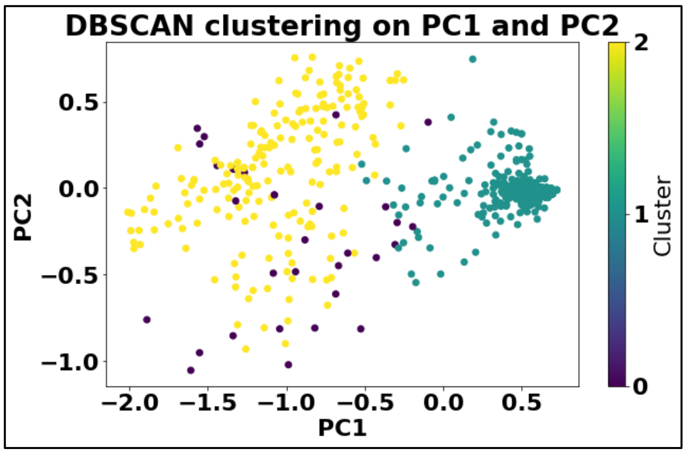

Figure 1. Scatter plot showing clustering results of DBSCAN, visualised using the top two PCs. Clusters are colour coded, with 0 representing the noise values.

Next, we use summary statistics to generate a high-level understanding of feature differences between tumour types (Table 2). We omit min and max values here as they show much overlap between clusters, however we do present the distribution of values for the Bare Nuclei feature later.

Cluster	Clump Thickness		UF Cell Size		UF Cell Shape	
	Mean	Std	Mean	Std	Mean	Std
Benign	3.12	1.90	1.34	0.79	1.47	0.98
Malignant	7.15	2.39	6.69	2.56	6.71	2.41
	Marginal Adhesion		**Epithelial Cell Size**		**Bare Nuclei**	
	Mean	Std	Mean	Std	Mean	Std
Benign	1.34	0.81	2.12	0.87	1.36	1.02
Malignant	5.68	3.02	5.24	2.25	8.59	2.32
	Bland Chromatin		**Normal Nuclei**		**Mitoses**	
	Mean	Std	Mean	Std	Mean	Std
Benign	2.10	1.12	1.30	1.00	1.12	0.76
Malignant	6.10	2.16	6.21	3.26	2.49	2.39

Table 2: Descriptive Statistics for discovered tumour clusters

From Table 2, we can see that malignant tumours have a larger mean value for all features than benign, and their higher standard deviation implies that there is a wider range of values that describe malignant tumours. Bare Nuclei, Uniformity Of Cell Size, and Uniformity Of Cell Shape show the largest difference in mean values between the tumour types (7.23, 5.35 and 5.24, respectively), and so may be the best features to focus on when classifying the tumour type. We next look at finding features that describe the most variance in the dataset, indicated by their absolute magnitude in the eigenvectors of the top two principal components from PCA.

Feature	PC1 Loading	PC2 Loading
Clump Thickness	0.30	0.07
UF Cell Size	0.40	0.23
UF Cell Shape	0.39	0.16
Marginal Adhesion	0.33	0.10
Epithileial Cell Size	0.25	0.00
Bare Nuclei	**0.44**	**0.78**
Bland Chromatin	0.29	0.01
Normal Nuclei	0.35	0.47
Mitoses	0.12	0.19

Table 3: PC 1 and 2 Loadings by feature

From Table 3, the Bare Nuclei feature has the highest loadings for both PC1 (0.44) and PC2 (0.78), which provided us with further evidence that this feature would be important for segmenting the two tumour types.

Cluster	Bare Nuclei Category	Frequency
Benign	≤ 5	172
Benign	> 5	59
Malignant	≤ 5	2
Malignant	> 5	450

Table 4: Distribution of Bare Nuclei Categories within Clusters

Finally, in table 4, we present the distribution of Bare Nuclei values by cluster. Notably, it's almost non-existent (only two samples) for a tumour to be malignant and have a Bare Nuclei value of less than 6. Therefore, measuring for these Bare Nuclei values might be a useful test to provide confidence in the benign nature of a tumour.

Conclusions.

We have demonstrated how to apply UL techniques to uncover hidden structures in a breast cancer dataset, as well as how to use this approach to learn about Malignant and Benign tumours. Our results concluded that the dataset was best clustered by DBSCAN at k=2 and highlighted the importance of the Bare Nuclei count in identifying tumour type. Interestingly, we note that Bare Nuclei had 16 missing values that were removed as part of our data cleaning. Given the importance of Bare Nuclei in understanding tumour type, future work should explore whether considering these missing values has any impact on the final feature analyses. One type of clustering approach which we did not consider was a hierarchical based technique, like the BIRCH algorithm [10]. It would be important to see whether this outperforms our current methods. During score validation, we noticed that our inaccurate scores for the GMM at $k > 2$ was likely due to a lack of regularisation and noise penalties that were implemented within the official scoring packages. We would hope to implement these features into our own scores, and improve their robustness, for future work. Finally, it would be beneficial to repeat our analyses on another breast cancer dataset, such as [11], as this would indicate whether the relationships we have discovered here generalise beyond this dataset. For example, [11] also contains a tumour size feature, and if its mean value was also higher for a cluster describing malignant tumours, it would provide confidence that our findings aren't just limited to this dataset.

References.

[1] W. William, UCI Machine Learning Repository, (1992).
[2] I. T. Jolliffe and J. Cadima, Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences **374**, 20150202 (2016).
[3] M. Patacchiola, [Internet] 2020.
[4] N. Sharma, [Internet] 2020.
[5] D. Arthur and S. Vassilvitskii, Society for Industrial and Applied Mathematics **SODA 07**, 1027 (2007)
[6] E. Martin, H. P. Kriegel, J Sander and X. Xu, Proceedings of the Second International Conference on knowledge Discovery and Data Mining **96**, 226 (1996)
[7] M. Halkidi and M. Vazirgiannis, Proceedings – IEEE International Conference on Data mining, ICDM , 187 (2001)
[8] Y. Liu, Z. Li, H. Xiong, X. Gao, J. Wu, IEEE International Conference on Data Mining, 911 (2010)
[9] P. J. Rousseeuw, Journal of Computational and Applied Mathematics **20**, 53 (1987)
[10] T. Zhang, R. Ramakrishnan and M. Livny, Data Mining and Knowledge Discovery **1**, 141 (1997)
[11] M. Zwitter and M. Soklic, UCI Machine Learning Repository, (1998)