CS7641 A3: Unsupervised Learning and Dimensionality Reduction

Charles Leung
cleung75@gatech.edu

I. Introduction

Unsupervised learning and dimensionality reduction are very useful tools that offer advantages over traditional supervised learning. Unlike supervised learning, unsupervised learning utilizes pattern recognition and structures within the data without the need for labels, making it very useful for data analysis and gaining insights. This can be a significant value add especially when labeling is expensive. Dimensionality reduction strategies such as Prinicipal Component Analysis (PCA) and Indepedent Component Analysis (ICA) transforms high dimensional data into lower dimensional form, while retaining paramount information about the dataset. This will mitigate issues with having high dimensions, such as creating more features and computations, and more unnecessary noise. In this exercise, we will explore the application of these techniques using two datasets from UCI's Machine Learning Repository - the Rice and Mushroom dataset.

A. Rice and Mushrooms (Datasets)

The rice dataset contains 7 total features: area, perimeter, major/minor axis length, eccentricity, convex area, and extent. All of these are numerical ratings that describe the physical features of the grain of rice. The output of this dataset is either an Cammeo, or an Osmancik grained rice. This means that we will be analyzing a binary classification problem, and our learner will only be predicting one of two outcomes. The objective now is to identify patterns within the dataset to cluster each piece of data based on the features described above.

B. Mushroom

The mushroom dataset contains 23 features and is more complex in nature. They are cap shape, cap surface, cap color, bruises, odor, gill attachments, gill spacing, gill sizes, gill color, stalk shape, stalk root, stalk surface above/below ring, veil color, ring number, ring type, spore print color, habitat, and population. The output of the dataset is either poisonous or edible, meaning this is another binary classification dataset.

II. Methodology and Hypothesis

As part of our investigation, we are going to use two clustering techniques to split our dataset: utilizing KMeans (KM), and Expectation Maximization (EM) clusterization. We will also be implementing dimensional reduction using PCA, ICA, and Randomized Projections (RP), analyzing them individually, and then putting the two concepts in conjunction to analyze the performance of our dataset. The reason why these datasets are interesting is because one of them is rather simple (rice), with highly correlated features to its targets, while the other being rather complex, with most features being neither highly or lowly correlated to its poisonous levels. It would be interesting to see the contrasting differences when performing these exercises.

Unsupervised learning techniques like KM and EM clustering partition datapoints into clusters. KMeans clustering utilizes distance metrics to calculate the euclidean distance from the K formed clusters. Expectation maximization uses Gaussian Mixture Models (GMM) clustering to make a probabilistic approach to clustering groups from Gaussian distribution matches and repeating this until convergence is hit (more details to be explained in the subsequent sections). Here, we will implement the elbow method, the Silhouette Score, and Adjusted Mutual Information (AMI) score to compare, contrast and assess the optimal cluster choices.

Dimension reduction techniques like PCA, ICA and RP are used to reduce complexity of our dataset by intricately divising features of the data so that its inherent attributes are preserved. By doing this, it will greatly increase efficiency of our dataset and reduce complexity to prevent things like overfitting. PCA utilizes linearity reduction techniques by performing orthogonal features called principal components. They are ordered by the amount of variance they capture from the data. ICA transforms data into components that are statistically independent from one another. Random Projections (RP), as the name suggests, utilizes random projections to transform higher dimension data to lower dimensions. In here, we will be utilizing the dimensionality analysis first, then inputting the simplified dataset into the clustering algorithm to compare and contrast the compute efficiency and compare its accuracy and results.

When applying these methods on both of these datasets, we will preprocess the data in the same method as A1, by encoding the mushroom dataset and not doing any preprocessing cleanup for either dataset. This way, we can provide consistent and accurate results when comparing unsupervised learning and supervised learning.

With this in mind, given that the neural networks performed extremely well in Supervised Learning, and

knowing that the rice dataset had distinctive, clean features (please see the correlation graphs listed in fig. 20) that heavily correlated to the target classification, we can hypothesize that unsupervised learning clustering should also provide relatively similar and accurate results. We can also hypothesize that the clean data with minimal noise will triage nicely and also increase its training and computational efficiency.

III. KM and EM Clustering

In this exercise, we will be analyzing the performance of Unsupervised Learning via KM and EM clustering in the Rice and Mushroom dataset. To do this, we will first use the elbow method to see when the number of components begins to decrease the rate of variance, it will indicate the best balance between clustering performance and model complexity and the subsequent optimal cluster number (K). The silhouette score and plot will also be generated to evaluate the quality of the clustering. By having these two metrics, we can analyze how receptive our data is to clustering.

A. Rice

Starting with the Rice dataset, we can first analyze the elbow method to see how the number of clusters influence the inertia in Figure 1 below. In this graph, inertia is defined by how well the data points are clustered around the centroids. In other words, it is a measure of how well teh data points are grouped around the centroids. It measures the spread of the data and retains the variance of the cluster. Here, the elbow method correlates to the point where adding more features does not significantly decrease the variance, as we want the model to be general enough to prevent overfitting, which is at around K = 4.

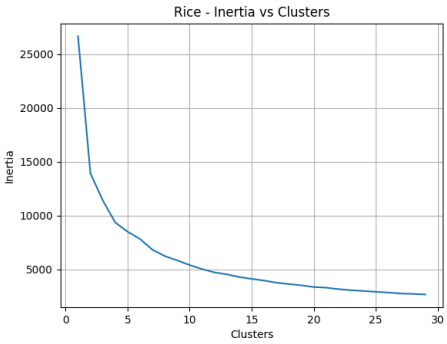

Fig. 1: Rice dataset - silhouette score from left to right. (a): silhouette plot per cluster, (b): clustering visualization.

From this, we can plot the silhouette plot and the subsequent cluster visualization, indicated by figure 2 below. Silhouette scores represent how distinctive each datapoint are categorized when comparing with other clusters. A score of 0 indicates that the datapoint is very close to the boundary from two neighboring clusters, and a score of 1 represents a definitively categorized datapoint. In this case, we can see that the silhouette score across all 4 clusters were relatively low at 0.27, with clear evidence from the cluster visualization in figure 2b that there were some overlap of different categories (outliers) that were detrimental to our clusters. This indicated that there should be more analysis needed to further define our dataset, like dimension reduction techniques. The low silhouette score is then agreed with the low/moderate AMI score at 0.41/1, indicating a moderate level of agreement between clustering results.

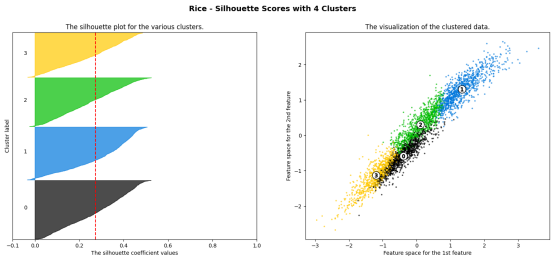

Fig. 2: Rice Dataset Inertia vs. Cluster size for Elbow Method analysis.

Next we move onto EM, where we will use the same methodology to determine our optimal cluster. First we will use the Bayesian Information Criterion (BIC) score as a metric to determine the optimal number of clusters between model fit and complexity. This will now be plotted against the number of clusters to determine the optimal number of clusters, calculated via the elbow method. From figure 3a, we can see that the optimal number of components is 4, as the diminishing returns from the score indicate that there is no strong incentive to increase the number of clusters (and thus the complexity of the model). Figure 3b then indicates the associated (surprisingly low) silhouette score of 0.17 and an AMI of 0.40. This suggests the clustering is of poor quality and further techniques are required to refine our clustering solution.

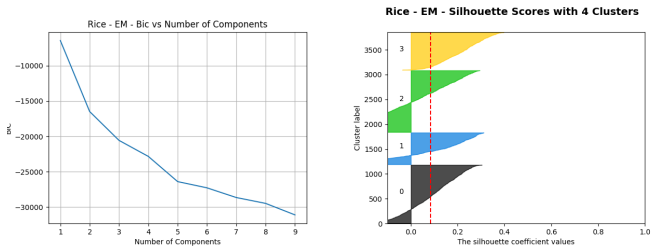

Fig. 3: Rice Dataset - EM. From left to right: (a) Inertia vs. Cluster size for Elbow Method, (b)Silhouette Score

B. Mushroom

In our mushroom dataset, the approach is identical to the rice dataset. Beginning with KM, we will analyze the elbow method, displayed in figure 4a. In here, we can see that the optimal cluster number is k=6, given that the rate of inertia decreases dramatically past that point. Interestingly, when we plot the silhouette score in figure 4b, it gave a low average of 0.32, and an AMI of 0.30.

Once again, the results show the clustering is not very strong, and little significance.

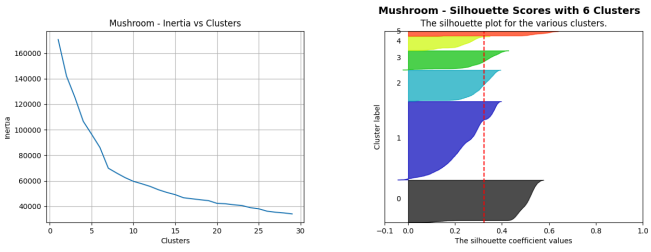

Fig. 4: Mushroom Dataset - KM. From left to right: (a) Inertia vs. Cluster - Elbow Method, (b) Silhouette Score

In the EM method, the elbow plot is plotted below via the BIC score in figure 5a, with the associated silhouette score in figure 5b. Here we determine the optimal cluster size is 8, and the associated silhouette and AMI score of 0.35 and 0.23 respectively. As expected, the results show overall weak clustering and and significance between clusters.

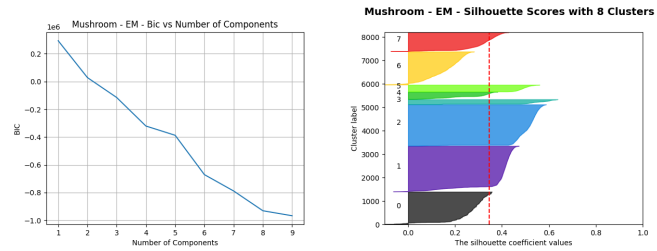

Fig. 5: Mushroom Dataset - EM. From left to right: (a) Inertia vs. Cluster size for Elbow Method, (b) Silhouette Score

C. Overall Observations/Conclusions

The low results in both KM and EM for both datasets can be because of the dataset's dimensions. In the mushroom dataset, for example, there is a high amount of features in the data, which could drastically increase the complexity of the model and can be hard to form clear clusters. To remedy this, performing a dimensional reduction technique can help alleviate the complexity and ultimately create easier cluster sizes. Additionally, there could be some overlap of classes in the data, which can hinder in creating discrete clusters. Finally, the features in the dataset itself may not be sufficient to capture the hidden complexities and patterns.

IV. Dimensional Analysis

In this section, we will implement PCA, ICA, and RP to reduce the number of dimensions in our two datasets. By reducing the nubmer of dimensions, we can ultimately reduce complexity and promote generalization/prevent overfitting. PCA maximizes the variance along the principal coordinates that are orthogonal to each other. ICA separates components into independent non Guassian components. RP reduces dimensionality by using random matrices to project high dimensional data to lower dimensional data and preserves the distance between datapoints with high probability. Finally, we will apply dimensional reduction for the two datasets and analyze the results.

A. Rice

We will begin with the Rice dataset, with the PCA analysis. Since we know that PCA preserves the number of variance, we will attempt to use variance to see how much of it is captured as we increase the number of components. From figure 6a, we can see that with 3 components, PCA will have already captured 100% of the variance, therefore we will reduce the number of components to 3. In ICA, the goal is to maximize non gaussian sources. Therefore, the item we want to maximize now is the kurtosis of the dataset. Recall, kurtosis is the measure of how disperse the data is between the distributions center and tails. From figure 6b, we can see the number of components to reduce to is around 5.

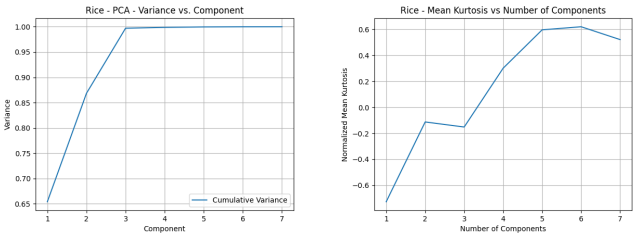

Fig. 6: Rice Dataset. From left to right: (a) PCA - Cumulative Variance vs. No. of Components, (b) ICA - Kurtosis vs. No. of Components

We can then implement the dimensional reduction using the three methods, indicated by figure 7 below. Note that even though some graphs suggested a higher number of component separation, we know that it is impossible to plot a higher order graph. Because of this, all three reductions are reduced to 3 dimensions. A future enhancement that could be made is to combine multiple dimensional reduction techniques together, and ultimately reduce the graph to the fewest dimensions possible. From the graphs, we can see that there is the clearest separation for PCA in dimensional reduction. This is perhaps because we have optimized the number of components to reduce to accurately from figure 6a. The slight underperformance in ICA could perhaps possibly speak to the dataset not aligning with non-guassianity and independence, while RP should be the worst in accuracy performance due to the randomized factor in projecting data into lower dimensional space (RP is useful when prioritizing computational efficiency only).

B. Mushroom

We will now repeat the same experiment for the mushroom dataset. Figure 8 describes the optimal component reduction value for both PCA and ICA via combined variance and maximized kurtosis. From the figure, we

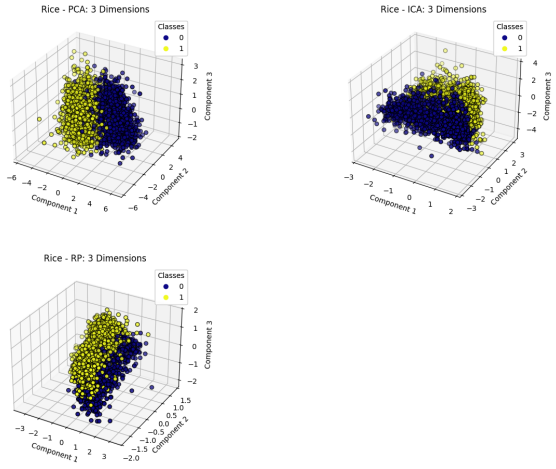

Fig. 7: Rice Dimension Analysis From top left to bottom right: (a) PCA, (b) ICA, (c) RP

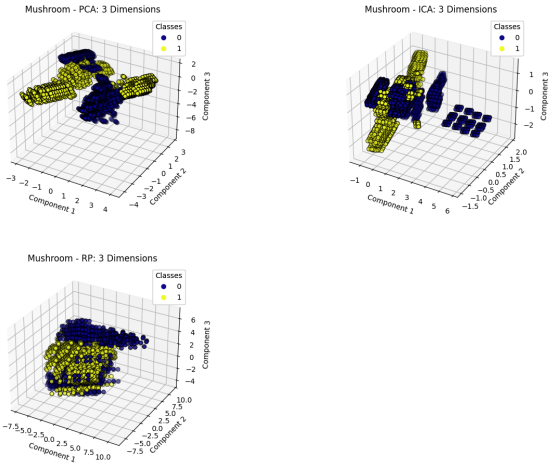

Fig. 9: Mushroom Dimension Analysis From top left to bottom right: (a) PCA, (b) ICA, (c) RP

can see that the optimal components for PCA is approx 15, as the number of captured variance when increasing the components become less significant and tapers off, suggesting that approximately 10 is a good attempt to try component reduction. In terms of ICA, the optimal components is around 4, as it yields the highest mean kurtosis and suggests a non gaussian behavior. The drastic increase in component requirements to capture the data's pattern speaks to how much more complicated the mushroom dataset is from the increased number of data features.

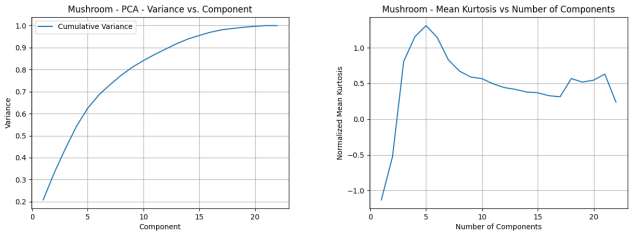

Fig. 8: Mushroom Dataset. From left to right: (a) PCA - Cumulative Variance vs. No. of Components, (b) ICA - Kurtosis vs. No. of Components

The three dimension reduction algorithms are now also applied to the dataset, and the results are displayed in figure 9. From the graph, we can see that there is a some separation, but it is not as defined compared to the rice dataset. The PCA graph here seemed to have scored the best in both the Rice and the Mushroom dataset, albeit still quite a bit of overlap. In addition, the RP and ICA graphs provide a much more murky result of some more overlap of classes and thus a less defined separation. The results here make more sense, as the complexity and many feature lists of the mushroom dataset is what's going to cause the model to have many intricacies, one that may not be caught with just three clusters.

V. Clustering with Dim. Reduction

In this section, we will use the two datasets we analyzed with, and perform dimension reduction techniques and followed by clustering technologies to find the optimal cluster number via elbow method and silhouette scores.

A. Rice

As mentioned earlier, we will first reduce the datasets dimensions into fewer items via PCA, ICA and RP. Then we will implement KM and EM clustering and discuss the results. Figure 10a-b showcases the elbow and silhouette plots for PCA + KM clustering, where as figure 10c-d showcases the same plots but for PCA + EM clustering. The AMI score from KM with PCA is 0.57, with a silhouette score of 0.41, while with PCA and EM, the AMI is 0.50, and the Silhouette is 0.35.

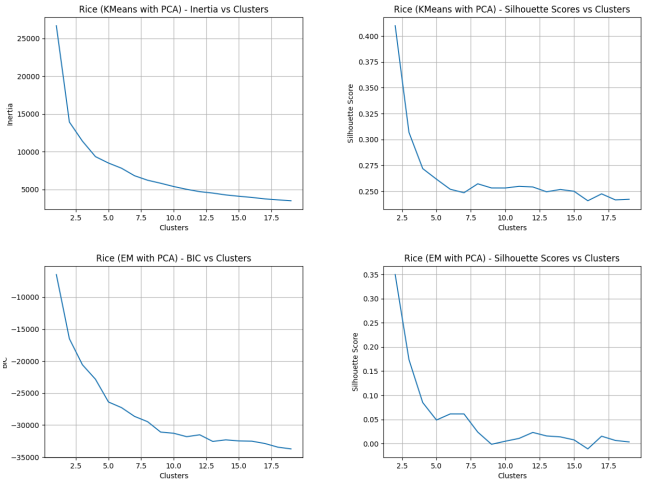

Fig. 10: Rice Dataset - Elbow and Silhouette Plots for PCA. From top left to bottom right: (a) Elbow Plot - PCA + KMeans, (b) Silhouette Plot - PCA + KMeans, (c) Elbow Plot - PCA + EM , (d) Silhouette Plot - PCA + EM

From the figure, we can deduce some critical and interesting information. To begin, figure 10a and 10c uses the elbow plot for KM and EM respectively, and both plots suggest that the optimal clustering value is 4 to capture most of the data's intricacies. However, the silhouette plot suggests that having 2 clusters provide the best separation and cohesion (quality of the cluster). Having a discrepancy of optimal clusters here is not unusual, as the elbow method focuses on the variance within clusters, where as the silhouette score measures how similar the features of the datapoints are with one another (quality of the cluster). Having 2 clusters for a binary classification problem inherently makes sense, which can possibly explain why a silhouette score is the highest for k = 2. The key here, however, is to understand the balance between the two analysis and choosing the right cluster value to prioritize accuracy or cluster quality.

Figure 11a-b now shows the ICA + KM behavior, where as figure 11c-d showcases ICA + EM, via the elbow and silhouette method respectively. Under this regime, the silhouette score yielded a maximum of 0.36 in KM and 0.32 in EM, while the AMI score was about 0.23 and 0.50 respectively.

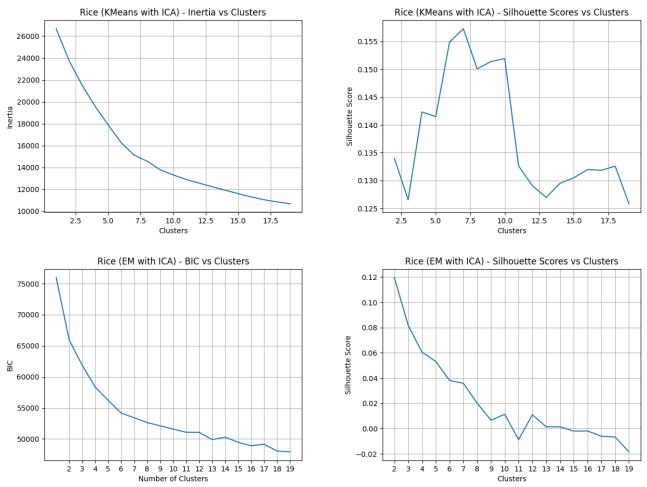

Fig. 11: Rice Dataset - Elbow and Silhouette Plots for ICA. From top left to bottom right: (a) Elbow Plot - ICA + KMeans, (b) Silhouette Plot - ICA + KMeans, (c) Elbow Plot - ICA + EM , (d) Silhouette Plot - ICA + EM

There is a notable spike in the silhouette score around 6-7 clusters for KMeans, indicating potentially better defined clusters at these points, contradicting the elbow method of returning 4 as the optimal cluster number. This is also evident in EM, where 2 clusters in the silhouette graph (11d) gave the highest score, where as the elbow graph returned 4. What we should do after is also investigate 6-7 cluster solutions to identify sub groups within the main class, which may provide some hidden information. Otherwise, we could also focus on the 2 cluster solution knowing that the dataset is binary in nature.

Finally, we are exploring RP with KM and EM respectively, displayed in figure 12. From here, we can see that KMeans provides a Silhouette score of 0.36, and an AMI of 0.41, where as in EM the silhouette score is 0.40 with an AMI of 0.60.

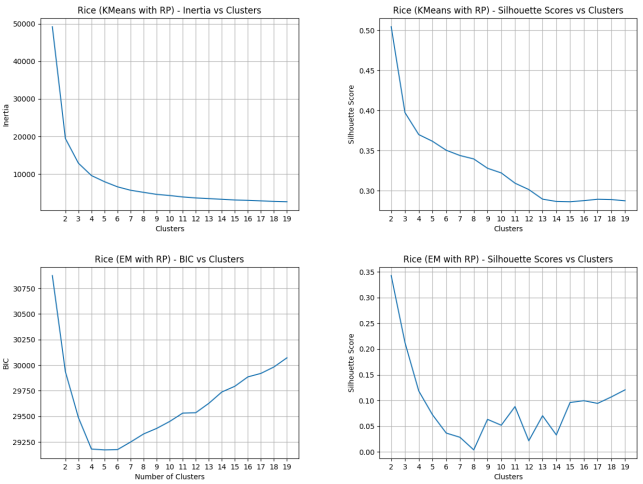

Fig. 12: Rice Dataset - Elbow and Silhouette Plots for RP. From top left to bottom right: (a) Elbow Plot - RP + KMeans, (b) Silhouette Plot - RP + KMeans, (c) Elbow Plot - RP + EM , (d) Silhouette Plot - RP + EM

Again we can see very similar results from ICA and PCA. From the elbow graph, it suggests to 4 being the optimal components. What is interesting though is the upward trend in clusters as we increase in figure 12c. This could be a sign of overfitting, since the number of clusters increases, causing the model to be more tailored to our specific dataset, which may not be the most optimal thing. Other than that, results have been pretty consistent in the silhouette score, with 2 being the overwhelming majority for silhouette school. This prediction aligns with the ground truth, since we know that these two clusters makes the most sense for ideal separation in a binary classification problem. Overall though, we can see that currently clustering is still showing suboptimal results, given from the lower silhouette score.

B. Mushroom

In the mushroom dataset, we will repeat all of our dimension reduction algorithms. Figure 13 will showcase the EM and KM for PCA. There we can see that the elbow methods in both EM and KM suggest that around 5 clusters is ideal, but the silhouette score suggests that 10 clusters provide the best defined and most meaningful cluster, with an associated silhouette score of of approx 0.36 for both KM and EM, and a AMI score of 0.25 for both KM and EM. An interesting concept is that there is a dramatic increase of the number of features between here and the rice dataset, speaking to the increased complexity of patterns present in the mushroom dataset.

Moving on, figure 14 showcases the ICA + KM and ICA + EM experiment results. What is interesting here is that ICA identified a much higher need for clusters

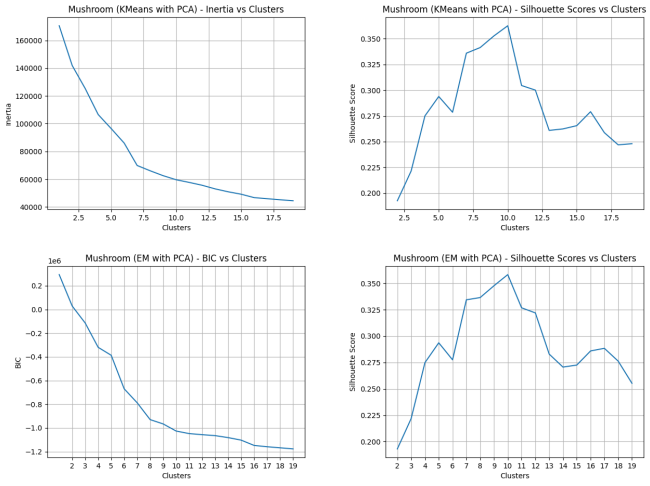

Fig. 13: Mushroom Dataset - Elbow and Silhouette Plots for PCA. From top left to bottom right: (a) Elbow Plot - PCA + KMeans, (b) Silhouette Plot - PCA + KMeans, (c) Elbow Plot - PCA + EM , (d) Sil. Plot - PCA + EM

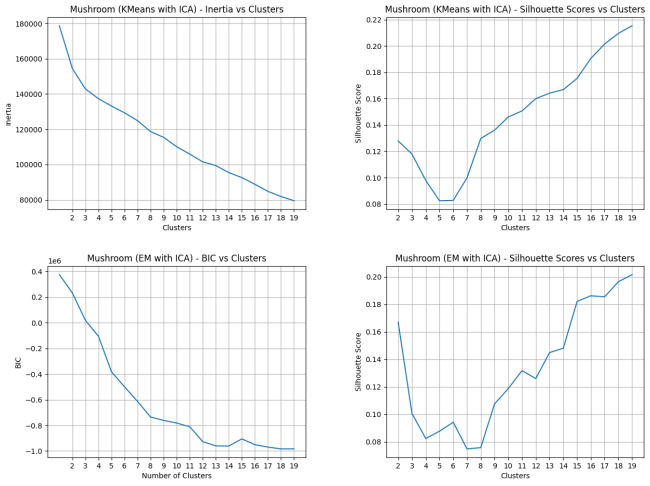

Fig. 14: Mushroom Dataset - Elbow and Silhouette Plots for ICA. From top left to bottom right: (a) Elbow Plot - ICA + KMeans, (b) Silhouette Plot - ICA + KMeans, (c) Elbow Plot - ICA + EM , (d) Silhouette Plot - ICA + EM

in the silhouette plot to produced the better results. This speaks to ICA needing more independent features, which results in the need for 19 clusters to find better statistically independent components. At this cluster, we can see the silhouette score generated is 0.18 and an AMI of 0.025 for KM, and a silhouette score of 0.24 and an AMI of 0.33. The elbow method also has mixed results. In KMeans (figure 14a), it seems like there is not a clear elbow to show a sharp decline in the the rate of inertia, but in EM (14c), we can see there is a clear taper at around clusters = 10. The low silhouette scores proves that it was a difficult feat to successfully triage the mushroom dataset using this dimensional reduction technique, suggesting that the dataset may not exhibit strong non gaussian characteristics that ICA handles well with, causing suboptimal performance.

Finally, we have RP and KMeans with RP with EM experiment results, depicted by figure 15. Beginning with KMeans, the figure suggests that the optimal clusters is 3 for both the elbow and silhouette plots. This yielded a silhouette score of 0.41 and an AMI of 0.02. Interestingly, for EM, the erratic behavior in the silhouette plot (15d) shows that it may not be well defined or consistent across different number of clusters. This could suggest that the nature of the randomized projection being a bit more spontaneous than other algorithms, particularly evident in figure 15d. Because of this, we can conclude that KMeans is more reliable with RP dimensional reduction techniques for the mushroom dataset. Because of its randomness, one enhancement that can be made is to rerun the experiment with multiple iterations to generate some variance and find the average, for a more accurate analysis.

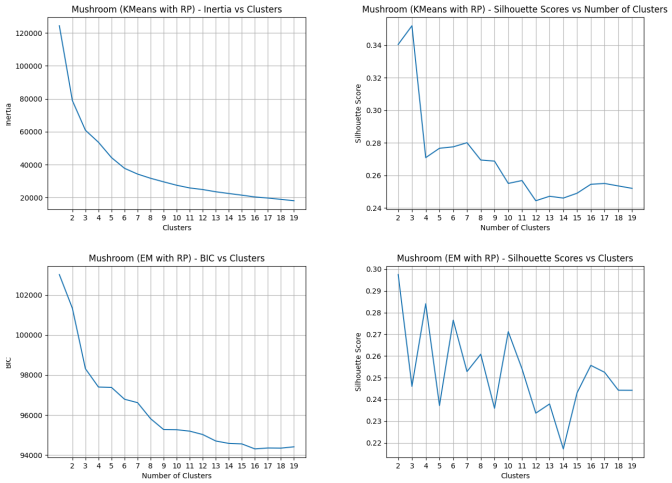

Fig. 15: Mushroom Dataset - Elbow and Silhouette Plots for RP. From top left to bottom right: (a) Elbow Plot - RP + KMeans, (b) Silhouette Plot - RP + KMeans, (c) Elbow Plot - RP + EM , (d) Silhouette Plot - RP + EM

VI. Neural Networks + Dimension Reduction

The approach for this section is to hone in on a specific dataset and perform dimensional reduction before training it with the neural network (NN). To do this and maintain consistency with the benchmark, we will first perform a gridsearch to find the optimal hyperparameters, and finally plot the associated learning curves to compare and contrast its performance with the benchmark MLPClassifier (via sklearn). These learning curves are constructed via a 5 fold cross validation so to reduce bias. In particular, we will analyze the nature of the dimension reduction tactics and how it may/may not impact the performance of the algorithm.

A. Benchmark

The benchmark in this case is the base neural network with all the dimensions preserved. Recall from A1, we

first investigated our hyperparameters and then tuned the using GridSearchCV. The hyperparameters and wall clock time are indicated from table I below. Using those parameters, figure 16 showcases the benchmark learning curve, which will be used to compare against the other three algorithms.

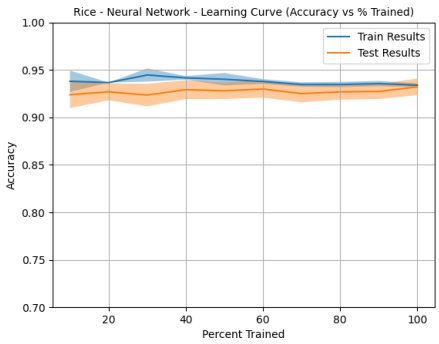

Fig. 16: Rice dataset - Neural Network Benchmark using MLPClassifier (sklearn).

B. Dimensional Analysis

Now we can dive deep into the dimensional analysis. For each of PCA, ICA and RP, we reduce the components and then perform hyperparameter tuning via GridSearch, feeding in our simplified dataset into the optimizer. This results in the optimized parameters in table 1, and the various learning curves displayed in Figure 17, with the hyperparameters and other meta data in table I below.

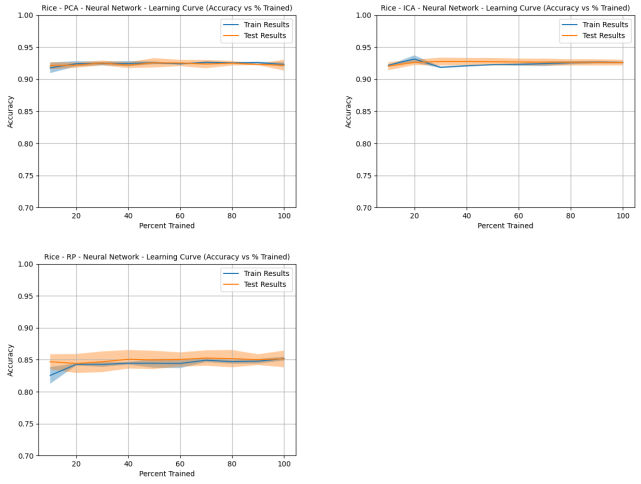

Fig. 17: Rice Dataset - NN with Dimensional Analysis. From top left to bottom right: (a) PCA, (b) ICA, (c) RP

Beginning our analysis with PCA (figure 17a), we can see that the provided graph matches and converges very nicely with minimal variance. This indicates that the model performed pretty optimally and was able to find the principal axes, capturing all of the hidden intricacies of the data, and ultimately orthogonalize the data and demonstrate its effectiveness. There is a slight gap between the training data and the test data, which may indicate that the learner may have been slightly overfit to be bias to the training data. However, it is obvious that the model proved to be as accurate as the supervised learner.

TABLE I: NN Hyperparameter and Performance Comparison - Dimension Reduction

Algo.	Act.	Alpha	Layers	Learn Rate	Train Time	Accuracy
PCA	tanh	0.01	2	0.1	0.25s	92.65%
ICA	relu	1.0	4	0.001	0.20s	91.86%
RP	relu	0.1	16	0.01	0.059s	88.19%
Bench	tanh	0.1	16	0.01	0.028s	92.4%

Following that with ICA (figure 17b), we can see that the graph also performs extremely well and captures the independent sources and the hidden patterns in the data features. Although there is convergence as the learner is being trained, indicating a well trained model that minimizes overfitting, we can see that the overall accuracy is less than PCA and also the benchmark at 91.9%. This suggests that the dataset assumes more of a gaussian like distribution in comparison to a non gaussian distribution, which makes sense as we know the dataset is fairly clean and simple to begin with.

RPs (figure 17c), by far performed the worst between the 3 learners, at 88%. This is the unfortunate result from the randomness and high variability from randomly projecting data to selected subspace, which ultimately losses essential information about the dataset. This can be seen from the larger variance bands in our test/train results in our figure. However, it compensates by offering the best computation efficiency compared to the other ones. Therefore, it ultimately depends on the user's preference: accuracy or speed.

Lastly, it is important to also compare the wall clock time to train the data. There is an increase in computational efficiency, from 0.29s to 0.25s in PCA, for example. This speaks to how powerful dimensional reduction truly is, as a simpler dataset will be multitudes less expensive to compute than a complex one.

VII. Neural Networks and Clustering

In the final section, we will implement and compare the learner against clustering techniques discussed earlier. The implementation strategy will be identical to the previous section, except with clustering instead of dimensionality reduction.

We will first perform GridSearchCV, and then finally use the tuned parameters to generate the the learning curve. The parameters are listed in Table II below, while KM and EM clustering graphs are described in Figure 18 and 19 respectively.

Beginning with KMeans Clustering, we first observe the histogram (fig 18a), which indicates there is a slight

imbalance between the distribution of the cluster. This could impact the the quality of the clustering, as we may have forced some datasets into a certain cluster group, impacting the quality of the cluster. With the calculation of the learning curve, we are able to see a nice convergence with a relatively high accuracy. Although the accuracy score is slightly lower at 89.4% compared to the benchmark at 92%. This could be attributed to the its inability to detect the underlying patterns as well as supervised learning. However, despite this, it still scored relatively close to our supervised learning benchmark. Therefore, we can safely agree with the hypothesis that our dataset's cleanliness created comparable results to the benchmark.

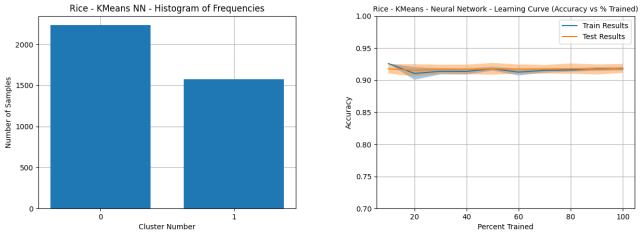

Fig. 18: Rice Dataset - Neural Network with KMeans Clustering. From top left to bottom right: (a) Distribution of Data (Histogram), (b) Learning Curve

TABLE II: NN Hyperparameter and Performance Comparison - Clustering

Algo.	Act.	Alpha	Layers	Learn Rate	Train Time	Accuracy
KMeans	tanh	0.0001	18	0.001	0.25s	89.4%
EM	tanh	0.01	18	0.001	0.50s	89.1%
Bench	tanh	0.1	16	0.01	0.028s	92.4%

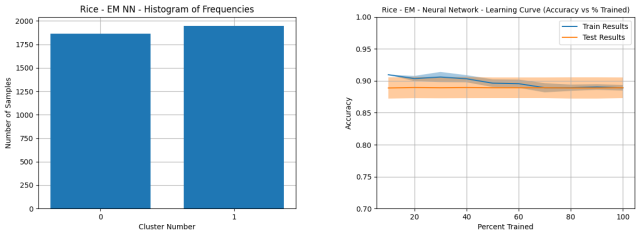

Fig. 19: Rice Dataset - Neural Network with EM Clustering. From top left to bottom right: (a) Distribution of Data (Histogram), (b) Learning Curve

Lastly, we have EM clustering (depicted in figure 19), which also performs similarly to KMeans, however the distribution of the clustering was a lot more balanced. This can be advantageous to avoid biases towards a class that appears more frequently. This also suggests that EM has effectively partitioned the dataset that can help in learning more effectively. In terms of the learning, the model's accuracy is relatively stable, and nicely converges with a small gap, indicating a minuscule amount of overfitting. Similarly to KMeans clustering, there is also a slight underperformance compared to supervised learning, with an overall accuracy of 89.1% compared to the benchmark of 92.4%. Despite this, the model was still extremely performant.

One thing to indicate is the wall clock time of training these algorithms. The KM algorithm took a smaller amount of time, given that the algorithm is relatively straightforward compared to EM, which is much more computationally intensive (requires computation of probabilities of each data point belonging to the cluster).

VIII. Comparison and Conclusion

Throughout this analysis, we were able to successfully explore the relationship between KMeans/EM clustering, dimensionality reduction, and the application of neural networks to both raw and dimensionally reduced datasets. Our initial hypothesis that the dataset would be performant in an unsupervised learning scenario was verified with our exceptionally converged learning curve, despite the supervised learner slightly outperforming by a slim margin. Our second hypothesis that the computation effiency was improved due to the dimensionality reduction. This was also generally verified, with a shorter computational speed to train the dataset in comparison to our benchmark. This can be important especially when the dataset grows to an enormous size - time and computation effort grows exponentially to be an all in all expensive task.

It is also good to relate our original feature correlations with our targets (shown in figure 20), and how it plays a crucial role how the results are the way they are. For example, having features that are highly correlations in the rice dataset is useful for PCA, as it is easy to extract the key features to orthogonalize and effectively reduce its dimensions. ICA, however, excels in extracting statistically independent features from one another. Because the mushroom dataset does not have dominating features that predict it's poisonous levels, it suggests that the dataset is rather gaussian in nature, and ultimately causing ICA to struggle in reducing its dimensionality.

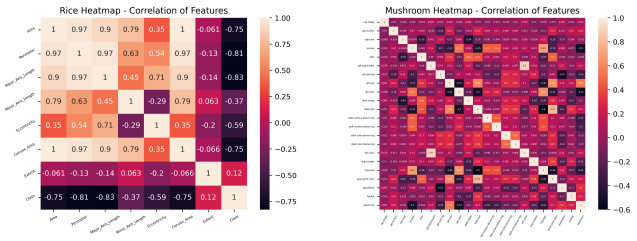

Fig. 20: Datasets correlation from left to right: (a) Rice, (b) Mushroom

To further enhance our exercise, we could also have combined dimensional reduction with a clustering method on our neural network to further compare its performance. We could also further verify silhouette scores and elbow methods by exploring the performance of it through multiple iterations so we can generate variance and reduce bias as well.