7642 Assignment 3 Report

Zebing Li
Georgia Institue of Technology
Atlanta, GA
zli738@gatech.edu

I. Introduction

This paper looks at clustering and dimensional reduction techniques. They are unsupervised learning algorithms where the algorithm is trained on unlabeled data. They are crucial in many real-world applications like data preprocessing, data visualization, etc. They can discover the hidden structures in data, improve the efficiency of algorithms, and provide meaningful insights. And this paper experimented with different clustering and dimensional reduction algorithms, and try to analyze the differences and suitability of them.

For clustering, I looked into K means and Expectation Maximization algorithms. And for dimensional reduction, I looked into Randomized Projections, Principle Component Analysis and Independent Component Analysis. I would apply the algorithms on two datasets in below sections. And I will train neural network on the outputs from the algorithms, and analyze their performances.

II. Dataset and Hypothesis

The two datasets used in this assignment are mushroom classification and heart failure prediction. They are from the assignment 1.

A. Mushroom Classification

It is a binary classification problem, giving different features of a mushroom and to predict whether it is edible or not. There are 9 features describing aspects of caps, gills, stem, etc. And the target class is edible (0) or not edible (1). Among the features, cap shape, gill color and stem color are categorical features, others are numerical. There are 5100 rows in this dataset.

This dataset is relatively imbalanced, 62% edible and 38% inedible. And this data doesn't have many outliers and noise. Also, the features follow quite distinct distributions. And the correlations between the features are relatively high.

B. Heart Failure Prediction

The dataset contains 11 attributes that are considered useful in predicting a possible heart failure death event, the follow up time and the target is whether the patient has died. Among the features, ejection fraction, diabetes, serum sodium and sex are binary, others are continuous. The dataset has 2500 rows.

This dataset has more noise and outliers than mushroom dataset. And it is also highly imbalanced, 70% non-death and 30% death. Besides, the correlation among the features are very small. And the distribution of the features are very different and non Gaussian.

C. Hypothesis

First, I think clustering would perform better in mushroom dataset. As there're noise and outliers in heart failure dataset, and the distances in clustering algorithms are sensitive to the outliers.

Second, I think PCA might perform better in mushroom, and ICA would perform better in heart failure. Because the PCA maximizes the variances that are explained, and I believe most information are obtained in variance for the mushroom dataset. And ICA tries to find independent sources, and I think the possible reasons causing heart failure death is likely to be independent.

D. Methods

The two datasets are split into 80% and 20% as training set and test set. I applied standard scaling to the features as they are of different scales and most of the algorithms are sensitive to scales. In neural network part, I used cross validation in training by equally splitting the training data to 5 pieces. The algorithms implementations are from Scikit-Learn package.

III. Clustering

A. Mushroom Dataset

First, I used K_means to do clustering on the dataset. A good clustering should minimize the differences between samples within the same cluster, and maximize the differences among different clusters. To measure the similarity of the same cluster, I used **Inertia** and **Silhouette score** metrics. **Inertia** measures the sum of squared distances of samples to their cluster center. A smaller inertia indicates closer distance to the cluster center and better clustering. **Silhouette score** compares the average distance of a sample to all other samples within same cluster with the average distance to all samples of neighbor cluster. Therefore, it measures how similar an object is to its own cluster compared to the other cluster. It ranges from -1 to +1, and the larger it is, the more similarities within the same cluster. On the other hand, to measure the differences among different clusters, I used the **Davies-Bouldin index** (DB index). It evaluates the similarity ratio of each cluster with its closet cluster. And a smaller db index indicates further difference between different clusters, and better clustering. These metrics can provide a comprehensive evaluation of how well K means has partitioned the data.

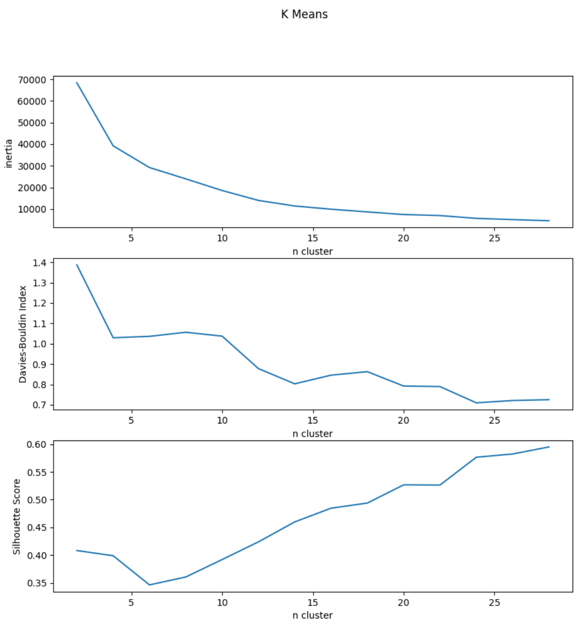

Fig. 1. K Means

1) K Means: For number of clusters, I tried number from 2 to 30, which should be a wide enough range. Looking at the plots **Fig 1**, first one plots inertia value against number of clusters. We can see as more clusters, inertia keeps reducing. This is reasonable as more cluster means more refined partitions and smaller distances between samples and centers. It shows a sharp decrease at the beginning, indicating that adding more clusters significantly increased the similarities between clusters, and improved the cluster quality. And the slope gradually flattens. After around 12 clusters, the rate of decrease slows down, suggesting diminishing returns on adding more clusters. According to "elbow" shape, 10 cluster might be a optimal candidate.

From the Silhouette score plot, it starts decreasing at beginning, then starts to increase shortly. The drop at beginning shows too few clusters don't perform well. As adding more clusters, the similarity between same cluster starts to increase indicating better performance. It keeps increasing at a high rate until around 20 clusters where it has some fluctuations and then increase again, and getting a highest score at end.

From the Davies-Bouldin index plot, we can see it drops steeply at beginning, flattens then starts to drop again at around 10. And from around 14, it starts to fluctuate and drop more slowly. The gradually dropping means the similarity between clusters keeps decreasing with more clusters, so the clustering quality is improving. Beyond 14 clusters, the benefit of adding more clusters starts to slowing down.

In conclusion, all three metrics show best performance at most number of clusters. However, around 14 clusters, the plots start to bend and marginal returns on adding more clusters begin to decrease. Considering the trade off between clustering quality and model complexity, I think **14 clusters** is an optimal number for K means.

2) Expectation Maximization: Since expectation maximization is probability based, and it is a soft clustering algorithm, the metrics to evaluate its performance are different from K-means and reply more on likelihoods. The higher likelihood indicates the better the cluster has explained the observed data. Also, similar with K means, the more similarity within same cluster compared to other clusters, the better the clustering quality. Therefore, I used **log likelihood, AIC, BIC and silhouette score** as metrics here. **Log likelihood** is straightforward and the larger it is, the better the clustering. **AIC** and **BIC** are similar as they are both a derived value from likelihood, and they also impose a penalty on large numbers of clusters. The larger the likelihood, and the more number of clusters, the smaller AIC and BIC and the higher quality of the clustering model. I think they are good metrics because they consider the trade offs between performance and model complexity, and prefer simpler models given likelihoods are the same. Same as K means, **silhouette score** is also used here to determine how similar a sample is to its own cluster compared with other clusters.

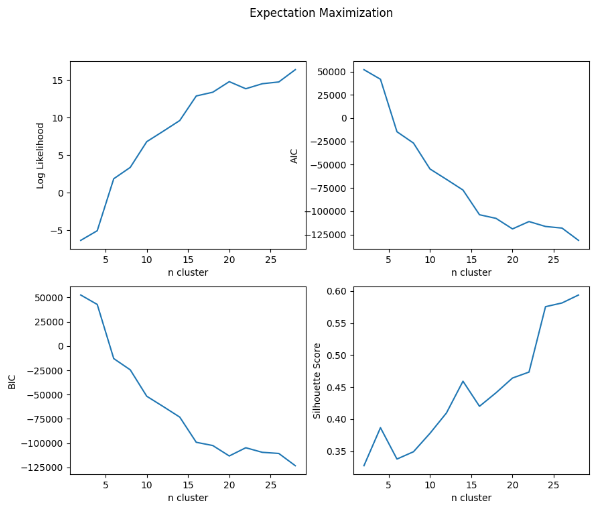

Fig. 2. expectation maximization

Now we look at the **Fig 2**. The x axis is number of clusters. I also chose from 2 to 30 which is same as K means to make the results comparable. First, the log likelihood increase steadily with the number of components, indicating that adding more components improves the model's fit to data. Also, after around 20 clusters, the increase starts to slow down and suggests decreasing effects. AIC and BIC are very similar as they are from similar calculations, with BIC imposes more penalty on large number of clusters. And they are almost the opposite of the log likelihood curve. The AIC and BIC keep decreasing until around 20 and it starts to fluctuate, then further decrease. This indicates the model improves as more clusters added. And after 20 clusters, the returns in adding more clusters start to diminish, as the increase in likelihood slows down and more penalty is added.

From Silhouette score plot, the value increases gradually, although there are some fluctuations. The score continues to rise till the end indicating that more components might still be beneficial in adding similarities in the same cluster. And at end, the score reaches to around 0.6 which is very close to the K means' score. So it appears that the two clustering algorithms are similarly effective in this dataset.

From the four plots, I think **20 clusters** is a good choice considering many curves starts to flatten out around 20. Although adding more clusters seem to continue have improvements, adding more components would increase the complexity of the model.

B. *Heart Failure Dataset*

Compared with mushroom dataset, the heart failure dataset adds more complexities. First, it has more features and more dimensions. From curse of dimensionality, it makes more difficult for the clustering algorithms to identify the distances and groups. Second, it has more noise and outliers. some clustering algorithms are very sensitive to outliers like Gaussian expectation maximization, because they can distort the distributions significantly.

1) K Means: First, we look at the K means performance. Same as the mushroom dataset, I still use Inertia, Silhouette score and Davies-Bouldin index as the metrics. Same number of clusters (2 to 30) are looked at here as I believe it is a wide range for this dataset as well.

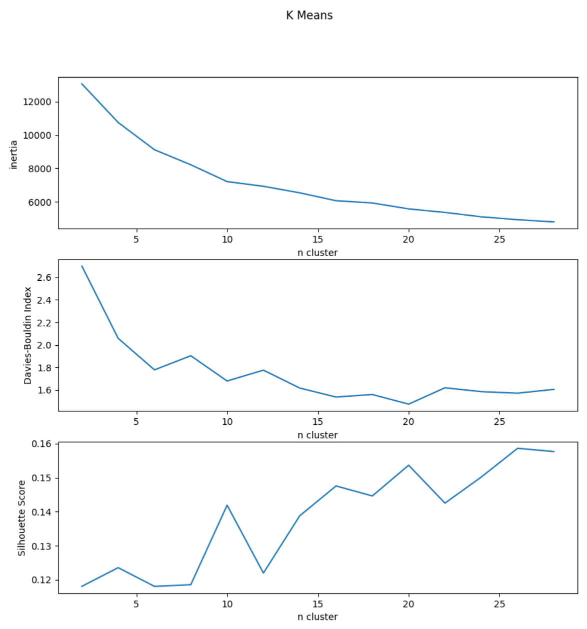

Fig. 3. K Means

Looking at **Fig 3**. First, from the inertia plot, the value keeps reducing as more clusters are used, indicating that adding more clusters improved the modeling quality. And the decreasing rate decreases gradually at around 10 and around 18, indicating a diminishing effect. From silhouette score, we can see that the curve is very fluctuating. I think this might be because of the outliers and noises of the dataset, sometimes it converges to a local optimal. But with more clusters, the score increased significantly, indicating the similarity between same cluster increases. The score reaches a relatively high point at about 18 and starts to decrease before increasing again. From the DB index plot, we can see it drops steeply at beginning, indicating the distance between different clustering decrease sharply. Then it flattens and drops again at around 10. And from around 16, it starts to fluctuate and starts to increase from 20. The effect of adding clusters starts to drop and adding more clusters at end eventually reduces the clustering quality.

Overall, I think **18 clusters** is an optima as the curves bend around this point and the changing rates start to decrease. However, comparing the final results with the mushroom dataset, this one only reaches silhouette score 0.16 at top, and mushroom dataset reaches 0.60. So this proves the **first hypothesis** is correct - the complexity of the dataset makes the K means perform worse.

2) Expectation Maximization: ' For expectation maximization, I still use log likelihood, AIC, BIC and silhouette score as metrics, and look at n clusters ranging from 2 to 30.

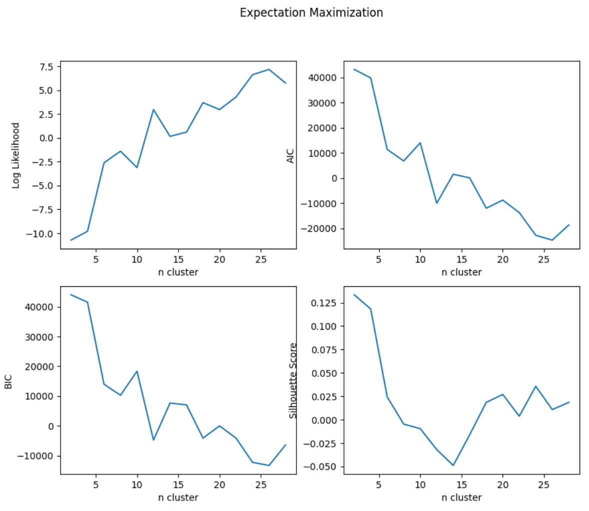

Fig. 4. Expectation Maximization

Looking at **Fig 4**. For log likelihood, it keeps increasing in general, although has some fluctuations in the middle. AIC and BIC are almost flipping the log likelihood curve, so it appears the likelihood is more dominant factor than the penalty applied in AIC and BIC. We can see at around 28, the likelihood reaches to the highest level, and then it starts to decreasing, indicating increasing number of clusters further doesn't help with model quality. Looking at silhouette score, it decreases very sharply until around 15 clusters. This indicates the similarity between same clusters are actually decreasing at beginning, although the likelihood is increasing. I think this probability is due to increased overlap or noise being interpreted as clusters. As number of clusters keeps adding, the silhouette score starts to increase at around 15 clusters. But the final score is only around 0.05, indicating the boundaries between clusters are not very separated, and some samples are close to the boundaries. From the four plots, I think **18 clusters** is a good optimal as the curves starts to fluctuate around the point, and the point overall has a good likelihood and silhouette scores.

3) Conclusion: Overall, comparing with mushroom data, expectation maximization performs worse for this dataset in all of these metrics. This is consistent with the first hypothesis. I think this is because the gaussian distribution is very sensitive to the outliers and noises which can distort the distribution. And comparing the expectation maximization with Kmeans, we can see expectation maximization perform worse in the silhouette score. I think this is because EM assumes the data can be captured by gaussian distribution, but here the dataset distribution is non Gaussian, which makes the EM struggle to accurately model the data.

IV. Dimension Reduction

A. Mushroom Dataset

1) PCA: To evaluate how well PCA works, I looked at the total explained variances, and reconstruction error. Since PCA finds components that projects most variances, looking at the **total variance explained** is important. Total explained variance is a sum of explained variance from each component. On the other hand, **reconstruction error** measures the difference between the original data and the reconstructed data, and indicates the information loss due to the dim reduction.

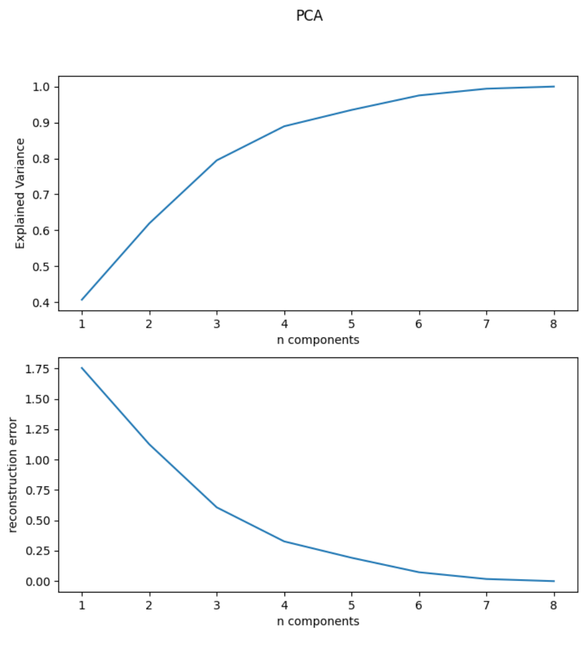

Fig. 5. PCA

The original data has 8 features. So I looked at number of components from 1 to 8. Looking at the plots **Fig 5**, the total explained variance keeps increasing as the number of components increases, as it captures more information from the original dataset. When the number of components grows to 8, it fully captures all variances as it is just rotates the original features. Reconstruction error keeps decreasing with more components, also indicating less information loss as number of components increases.

From the plot, the increase rate for explained variance and decrease rate reconstruction error both begins to decrease at 4 components, where total explained variance is about 90% and reconstruction error is about 0.4. Although with more components, they can further improve but it makes the model more complex. So I think **4 components** is a optimal number to choose.

2) ICA: For ICA, I chose **kurtosis** and **reconstruction error** to evaluate the performance. First, ICA assumes the underlying sources are independent, and it targets to find the components that are maximally non-Gaussian because they are more likely to represent the independent underlyings. And kurtosis can measure the non-Gaussian of the components. The more positive or more negative the kurtosis is, the tail is more heavy or more light than Gaussian, and therefore is more non-Gaussian. And the more it's close to zero, the distribution is closer the Gaussian. The reconstruction error is same as in PCA which measures the information loss.

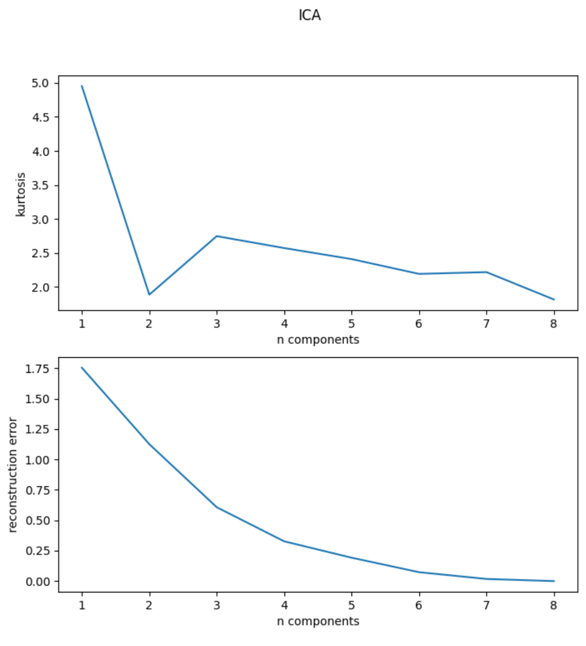

Fig. 6. ICA

From **Fig 6**, the kurtosis plot shows the averaging kurtosis across components. We can see the value drops sharply from +5 to +2 at begining, indicating the Gaussianity is increasing when adding the second component. Then it increases and stabilizes with minor fluctuations. I think this suggests the first few components capture most of the non Gaussian structure, as increasing more components after three doesn't increase the kurtosis. And the kurtosis is not very large, so this indicates that ICA might not be a good algorithm here as the sources are not really independent. Reconstruction error keeps decreasing which is same as PCA, as more components includes more information.

Based on the kurtosis and reconstruction error plots, a reasonable choice for the number of independent components seems to be **4**. This is where the reconstruction error significantly decreases and kurtosis stabilizes, indicating that most of the important non-Gaussian structure and significant information in the data are captured.

3) Randomized Projection: For randomized projection, I looked at **reconstruction error** and **pearson correlations**. Since randomized projection uses random matrices to project original data, it is important to look at how well it preserves the original structure and information contained in data. Reconstruction error measures the information loss by looking at the error in reconstructing the original data from projected data. And pearson correlation accesses how well the correlation between data points are preserved after the projection. They together well evaluates the effectiveness of RP.

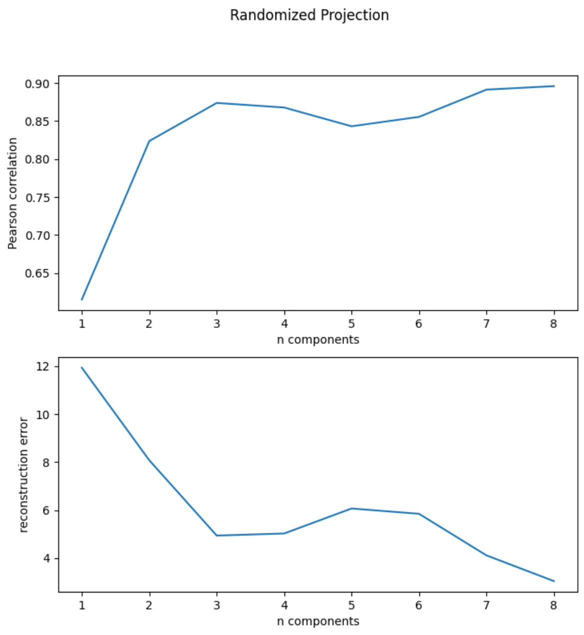

Fig. 7. Randomized Projection

Looking at **Fig 7**, from reconstruction error plot, the error decreases as the number of components increases. Initially, there is a significant drop in the reconstruction error from 1 to 3 components, indicating a better approximation of the original data. Beyond 3 components, the error reduction is less dramatic but continues to decrease, showing a consistent improvement in the quality of the reconstruction as more components are added. However, compared with PCA and ICA, reconstruction error is larger for RP, as RP use random matrix without assuming any structure of the data. But the decreasing error still indicates that it is very effective. From pearson correlation plot, as the number of components increases, the Pearson correlation increases and stabilizes around 0.85, indicating the structure is better preserved with more components. And the correlation is relatively high from 3 components onward, suggesting that even with a small number of components, the projection preserves the data structure well.

Overall, **3 components** seems to be a likely optimal for randomized projection. As both the reconstruction error and pearson correlation starts to stabilize, and the data structure and information are relatively well preserved here.

B. Heart Failure

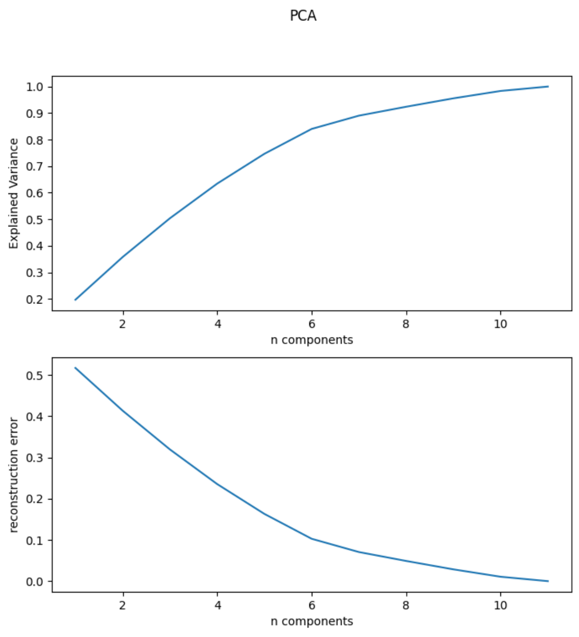

Fig. 8. PCA

1) PCA: This dataset has 11 features, so I looked at the n components for dim reduction within range 1 to 11. Looking at Fig 8, **Total explained variance** keeps increasing with more components. And the slope is sharp at beginning, because first few components capture most variances. And later components explains less variance. The curve starts to flatten around 6 where adding more components have diminishing effects. For **reconstruction error**, it keeps decreasing as more components contains more information and reduce the error. Similarly, the curve starts to stabilize at around 6 components.

In conclusion, **6 components** is an optimal point, as it captures over 80% total variance, and error to reconstruct original data is only 10%. Adding more components have less marginal effect, so considering the trade off from model complexity, 6 components look ideal.

2) ICA: For ICA, I looked at kurtosis and reconstruction error. Looking at Fig 9, **Kurtosis** grows from around +2 to +10

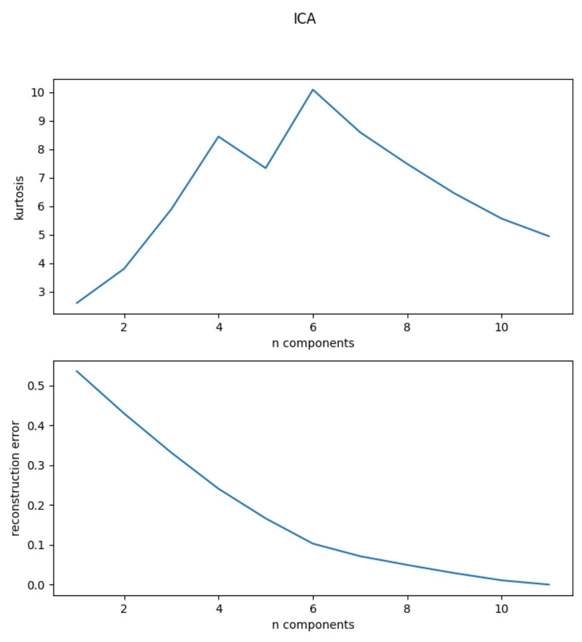

Fig. 9. ICA

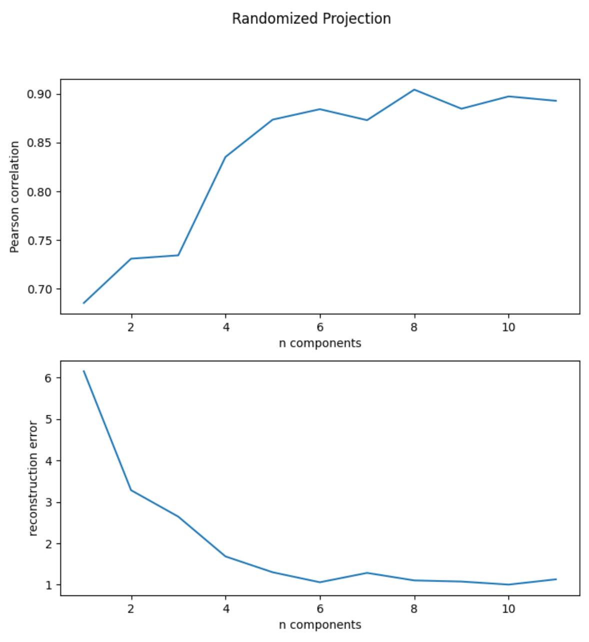

Fig. 10. SRP

from 1 components to 6 components, indicating adding more components improves the separation of independent sources and results in more non-Gaussian components. Beyond 6 components, the decrease in kurtosis suggests that additional components are less independent. **Compared with mushroom dataset**, kurtosis is generally larger here, indicating ICA is more effective here and the sources are more independent. **Reconstruction error** decreases consistently, starting with steep drop and then slows down at around 6, indicating adding more components doesn't capture as much information as earlier components.

Overall, **6 component** is an optimal for ICA as well. It reaches kurtosis=10, indicating the components are highly non gaussian, and reconstruction error is as low as 0.1. Beyond this point, the two curves start to bend or stabilize, indicating diminishing returns. Given the results, ICA is very effective in this dataset, indicating the underlying sources are more independent.

3) Randomized Projection: We again look into the reconstruction error and pearson correlation for randomized projection. From Fig 10, for **reconstruction error**, it decrease significantly as number of components increases to 6, showing more information are gained with more components. After 6, it starts to stabilize at around 1 and indicates a less significant effect of adding new components. Pearson correlation gradually increases and stabilizes at around 85%, indicating the correlation and structure of original data is well preserved.

Overall, **6 component** is an optimal for RP as well, where reconstruction error and pearson correlation both starts to stabilize. It reaches reconstruction error =1, and pearson correlation = 88%. So it shows that RP can effectively preserve the information and structure of the data, and is very fast compared to ICA and PCA in the meanwhile due to its randomized feature.

V. Dimension Reduction then Clustering

A. Mushroom Dataset

Now we run dimension reductions first, then apply clustering. When choosing the n_clusters and n_components, I used the optimal got from previous steps. N_clusters = 14 for K means, and 20 for expectation maximization. N_components = 4 for PCA and ICA, 3 for randomized projection.

	Original Data	PCA	ICA	SRP
Inertia	11447	5268	1512	3595
DB Index	0.80	0.57	0.67	0.79
Silhouette Score	0.46	0.58	0.52	0.50

TABLE I
K Means

1) K Means: The **table 1** uses the metrics for clustering algorithms, in order to compare how clustering performs after dim reduction. First looking at K means, PCA resulted in the highest inertia, indicating less tight clusters compared to ICA and SRP. It has the lowest db index, suggesting that clusters are more well-separated, and the highest silhouette score suggesting better defined and well separated clusters. ICA has lowest inertia meaning it has the tightest clusters. And its db index is higher and silhouette score is lower than PCA, indicating the clusters are less well separated and well defined. SRP has intermediate inertia, and highest db index and lowest silhouette score. This indicates that the clustering quality is not as good as the other two algorithms. Comparing with the original data, all of the three algorithms have significantly improved the clustering quality. Therefore, dim reduction is very helpful in K means.

In conclusion, PCA seems to be the best choice for preprocessing before K-Means clustering in this scenario, as it provides the best cluster separation and definition, which are crucial for meaningful clustering outcomes. ICA might be preferable if the focus is on having very tight clusters, while SRP seems less effective in this case. But all of the three are helpful preprocessing techniques compared to original data.

	Original Data	PCA	ICA	SRP
Log Likelihood	14.80	1.09	3.96	-1.84
BIC	-113278	-6391	-29805	16651
AIC	-118954	-8279	-31693	153954
Silhouette Score	0.53	0.62	0.53	0.52

TABLE II
EXPECTATION MAXIMIZATION

2) Expectation Maximization: From **table 2**, for log likelihood, original data has largest likelihoods, indicating the best model fit compared to the dim reduced data. ICA has the largest log likelihood with the three algorithms, suggesting better model fit. RP has the least likelihood, suggesting the worse model fit. For BIC and AIC, original data has much smaller values indicating best model fit, followed by ICA, then PCA and SRP. For silhouette score, PCA has the highest score, suggesting better defined clusters. ICA, SRP and original data have similar scores.

So in conclusion, for EM, dimension reduction seem to reduce the model fit for clustering. This might be because after dim reduction, the features are rotated and more non gaussian, so it is not modeled well in GMM. But the clustering quality seems to be improved largely by PCA. Therefore, I think PCA is the best choice among the dim reduction algorithms for EM.

In conclusion, I think best combination is **PCA + K means**, which is consistent with the **second hypothesis**. As it significantly improved the clustering quality in separation compared with other algorithms as well as the original data.

B. Heart Failure Dataset

I used the optimal got from previous steps for the heat failure dataset. N_clusters = 18 for K means, and expectation maximization. N_components = 6 for PCA , ICA, and RP.

	Original Data	PCA	ICA	SRP
Inertia	5933	3615	3566	4680
DB Index	1.56	1.24	1.23	1.37
Silhouette Score	0.14	0.21	0.20	0.18

TABLE III
K MEANS

1) K Means: From table 3, first looking at K means, ICA and PCA have very close results. ICA resulted in the least inertia, followed by PCA, indicating more tight clusters compared to SRP. ICA and PCA also have the lowest db index, suggesting that clusters are more well-separated. They are also have the highest silhouette score suggesting better defined and well separated clusters. SRP have worse results in all of the three metrics compared to ICA and PCA. This indicates that the clustering quality is not as good as the other two algorithms. Comparing with the original data, all of the three algorithms have significantly improved the clustering quality. Therefore, dim reduction is very helpful in K means.

Overall, ICA seems to be the best choice for preprocessing before K-Means clustering in this scenario. PCA is very close too, while SRP seems less effective in this case. But all of the three are helpful preprocessing techniques compared to original data.

	Original Data	PCA	ICA	SRP
Log Likelihood	3.69	-5.24	-5.28	-6.64
BIC	-4112	24770	24957	30358
AIC	-11969	21953	22140	27541
Silhouette Score	0.14	0.09	0.08	0.05

TABLE IV
EXPECTATION MAXIMIZATION

2) Expectation Maximization: Looking at table 4, original data outperform in all four metrics – it has largest log likelihood, smallest ais and bic, as well as highest silhouette score. Therefore, the dim reduction doesn't perfom really well for EM. Comparing the three algorithms, PCA is the best performer as the log likelihood and silhouette score are larger than the other two. However, the likelihood is negative, and the score is merely 0.09. So in general, I think EM perform poorly for this dataset, due to the non-Gaussian feature and existence of noise and outliers.

In conclusion, I think best combination is **ICA + K means**, which is consistent with the **second hypothesis**. This indicates that the source for the features have independence and can be modelled well by ICA. And K means seems to be more robust for the noise and outliers of the dataset. Also, GMM assumes the samples come from gaussian distributions which is not suitable for this dataset.

VI. NEURAL NETWORK ON DIMENSION REDUCTION

This section, I use the mushroom dataset. First, I preprocess the data using dim reduction algorithms and then train the neural network and compare with the original data. I use f score as the metric to evaluate the neural network, as it is a good metric for unbalanced data.

Looking at **fig 11**, original data has the highest F1 scores (0.95) for both training and validation, indicating the best model performance. The training and validation curves are very close, and stabilize at end, indicating minimal overfitting and good generalization. PCA reaches a relatively high f1 score at around 0.85. And the training and validation curves are close and flattens at end, indicating good generalization. ICA gets a lower F1 at around 0.66. And the learning curves is not stabilizing at the end of the training, indicating that it

	Original Data	PCA	ICA	SRP
Training Time	2.67	2.53	2.50	2.50
F1 Score	0.96	0.84	0.80	0.77

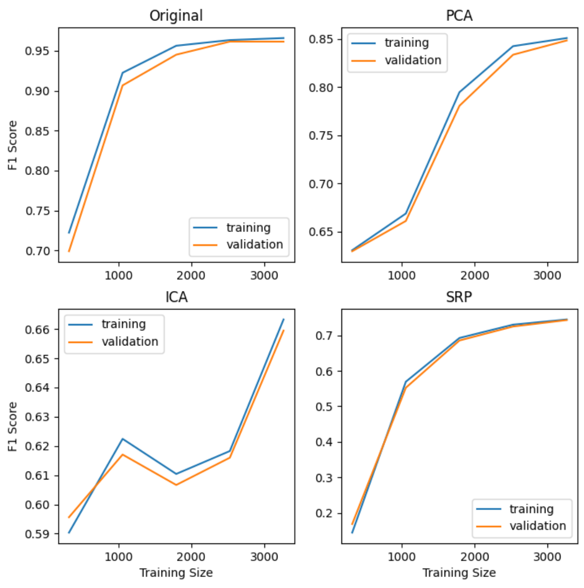

Fig. 11. Neural Network

	Original Data	K Means	EM
Traning Time	2.67	2.66	2.93
F1 Score	0.96	0.97	

might need larger training size to converge. SRP also gets a F1 score at around 0.66, but its training and validation curve show it generalize well as they are very close and stabilize at end.

Looking at **table 5**, neural network trained on dim reduction method can reduce the training time, since the data dimension is reduced. But F1 score clearly has decreased compared to original data, and PCA performs the best in the three algorithms.

In conclusion, the reduced data in general reaches smaller f1 score comparing with the original data, because they lost some information during the projection. But in the meanwhile, they can reduce the training time. Among the dim reduction, PCA reduces the dimensionality while maintaining relatively high model performance and good generalization. ICA and SRP however lead to a decrease in model performance.

VII. Neural Network on Clustering

For clustering algorithms, there are some additional processing on the output before training the neural network. I take the output from the clustering, and create number of cluster additional flag variables with values 0 or 1, where 1 indicating the sample belongs to the particular cluster. Then combine the flag variables with the original data, and feed it into neural network training.

First, looking at the training curves **Fig 12**, we can see the clustering algorithm starts at a higher f1 score then original data. I think this is because clustering provides additional

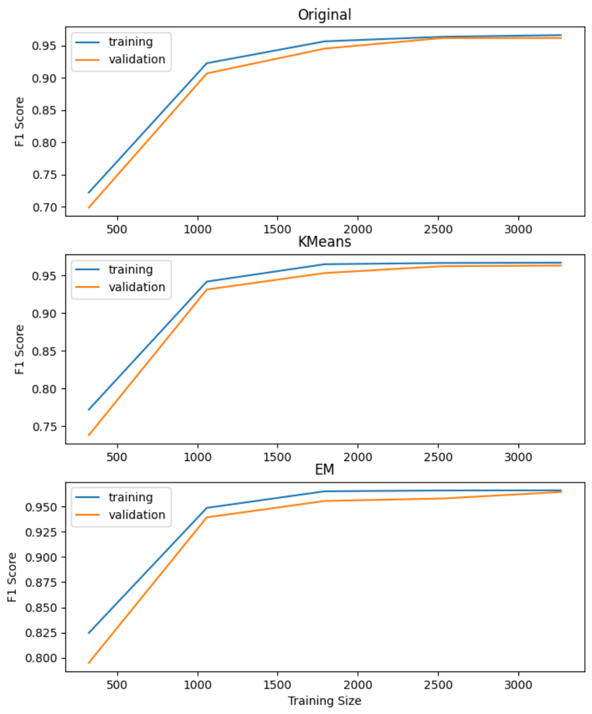

Fig. 12. Neural Network

information and this helps the neural networks to set up and learn. And they all reach a high F1 score at around 0.96 at end, and generalize well with minimal overfitting.

Then we look at the **table 6**, since the two clustering algorithm adds more features to the original dataset, it can increase the training time of neural network. We can see K means almost takes same amount of time, and EM takes more time as I used more clusters in EM. For F1 score, K means achieves the highest score indicating that it improves the neural network performance. And EM reaches as high F1 score as the original dataset.

In conclusion, clustering can help improve the model accuracy in classification. As a trade off, it might increase the training time as it introduces new features to the original dataset.