Unsupervised learning and dimensionality reduction

Zhihua Jin
zjin80@gatech.edu

1. Introduction

In this report, I studied the unsupervised learning algorithms and dimensionality reduction algorithms. First, I applied clustering on the data points to find; Next, I extracted features from the original data by dimensionality reduction, and applied clustering on the data points. This will tell us whether the dimensionality reduction algorithms extract the useful features successfully or not. Visualization on 2-D space gives the intuitive presentation. Lastly, I used the clustering result to assist the supervised learning task.

Datasets are the same as my 1st homework. The first one is "UCI Credit Card" dataset (Kaggle.com, 2019). Dataset 1 has 30000 samples, each of which is categorized into 24 features and 1 target variable. The second dataset is "Gender Recognition by Voice" (Kaggle.com, 2019), in which consists 3000 pre-processed voice samples and 21 variables. The interesting part is that the voice dataset is a small dataset with 50/50 classification problem while the credit dataset is an imbalanced 80/20 distribution dataset with samples 10 times more. This time, I did not adopt feature engineering. Also, I normalized the feature into the range [-1, 1] for dataset 1 for the correctness to calculate Euclidean distance regarding in clustering algorithms.

2. Clustering

a) K-Means

K-Means algorithm starts from k clusters, each with a random center. Then each data point is assigned to its closest center, and the centers are re-calculated by averaging the data points in a cluster. By repeating doing this, the assignment of data points will become stable when it reaches convergence. Denoting the data points as x_j, and the cluster center as $x^{(k)}$, I used 3 metrics that can be used to evaluate the clustering results. There are certainly other metrics such as mutual_info_score and completeness_score, etc., but some of them require the help of "label-true". Considering the evaluation universality since we are doing unsupervised learning, I prioritized those metrics that do not need labels. They can all be implemented using scikit-learn (Scikit-learn.org, 2019)

- SSE: This is sum of square error of each data points to its cluster center. $\text{SSE} = \sum_k \sum_{j \in C(k)} \left| x_j - x^{(k)} \right|_2^2$. The smaller the within-cluster SSE, the similarity between objects in that cluster is higher .
- CH-score: Calinski Harabasz Score measures the compactness of the clustering by calculating the sum of squares of distances between each point and the center within-cluster, and measures the dispersion between clusters by calculating the sum of squares of distances between various center points and the center of the data set. It is obtained by the ratio of separation degree and compactness (Scikit-learn.org, 2019). Thus, a larger CH means the optimal clustering results. The equation is $\text{CH} = \frac{\sum_k a_k \left| x^{(k)} - \bar{x} \right|_2^2}{\text{SSE}}$.
- Silhouette Coefficient: It is another metric to interpret consistency within clusters of data and examine how well data points are classified. The Silhouette Coefficient for a sample is

(b - a) / max(a, b), in which a is the mean in-cluster sample distance and b is the distance between a sample and its nearest cluster the sample is not inside (Scikit-learn.org, 2019). The value ranges from -1 to 1. With value 1, the clusters are likely to be classified correctly. If the value is -1, then it indicates that the specific sample should have been classified into another cluster.

- F1-score: In addition, I used F1-score to supplement the analysis. Once a pair of data points with SAME label are assigned into the same cluster, it is a true-positive; if they are assigned into different clusters, it is a false-negative. Once a pair of data points with DIFFERENT label are assigned into the same cluster, it is a false-negative; if they are assigned into different clusters, it is a true-negative. This is calculated by self-implement function.

Take dataset 1 as example, by varying *k* from *2* to *10*, the metric curves are shown below. Using elbow method (the inflection point), I observed a quick decrease when *k* changed from *2* to *5* on the SSE curve. Increasing *k* after that cannot result in much better improvement. Also, judging from the CH score figure, when *k>5*, there is a rapid decrease. So I could determine the best value in the range [3,4,5]. While in the F1 curve, f1 score decreases rapidly when *k=5*.

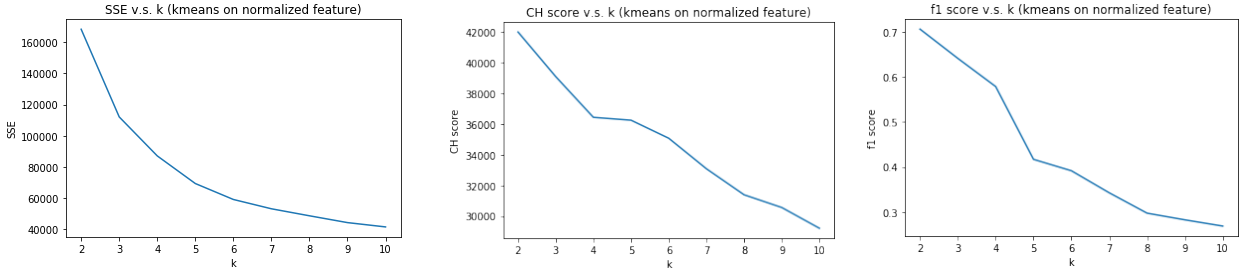
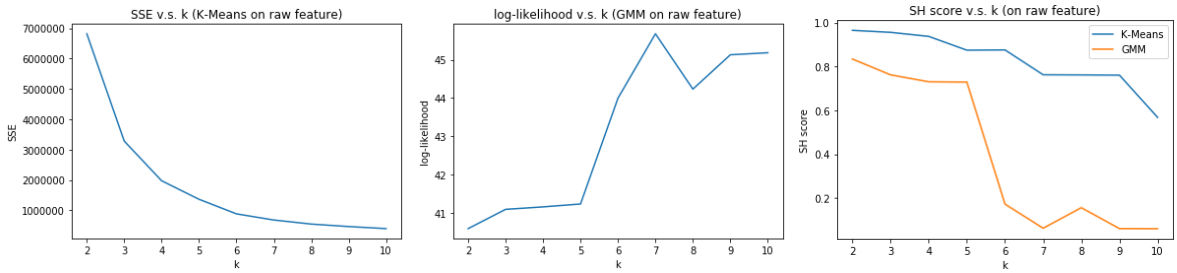


Fig. 1 Metric curves for K-cluster: Dataset 1

Looking at the above factors and silhouette method, I determine the final optimal choice for dataset 1 to be k*=4. Similarly, the optimal k for dataset 2 is 6 (Fig 2).

b) Gaussian Mixture Model (GMM) and Expectation Maximization (EM)

Instead of SSE, GMM algorithm estimates the likelihood of each data points. It is trained with the Expectation Maximization (EM) algorithm. For EM training, it judges whether a model fits well by observing the probability value of sampling and the model. Then the model is adjusted to fit the probability value of the sample. This process is iterated many times until the two probability values are very close. The first metric I used is log-likelihood of all data points. It helps find the optimal parameter since it will maximize the log-likelihood (Taboga, 2019). The other metrics remain the same. The figures below show the metric curves of dataset 1 as *k* varies from 2 to 10. In the log_likelihood curve, it shows little improvement when increasing *k*, while the other two curves show great decrease.



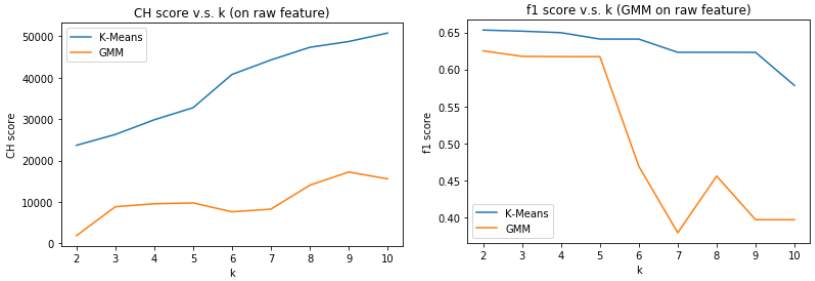


Fig. 2 Metric curves for K-cluster & GMM: Dataset 2

Considering these findings, I determine the final optimal choice for dataset 1 to be k*=2 and dataset 2 to be 5. Similarly, the optimal k for EM algorithm in both datasets are set.

Here follows the clustering result with t-SNE as 2D visualization tool (En.wikipedia.org, 2019). We can see the clustering effect of K-Means and GMM on the datasets.

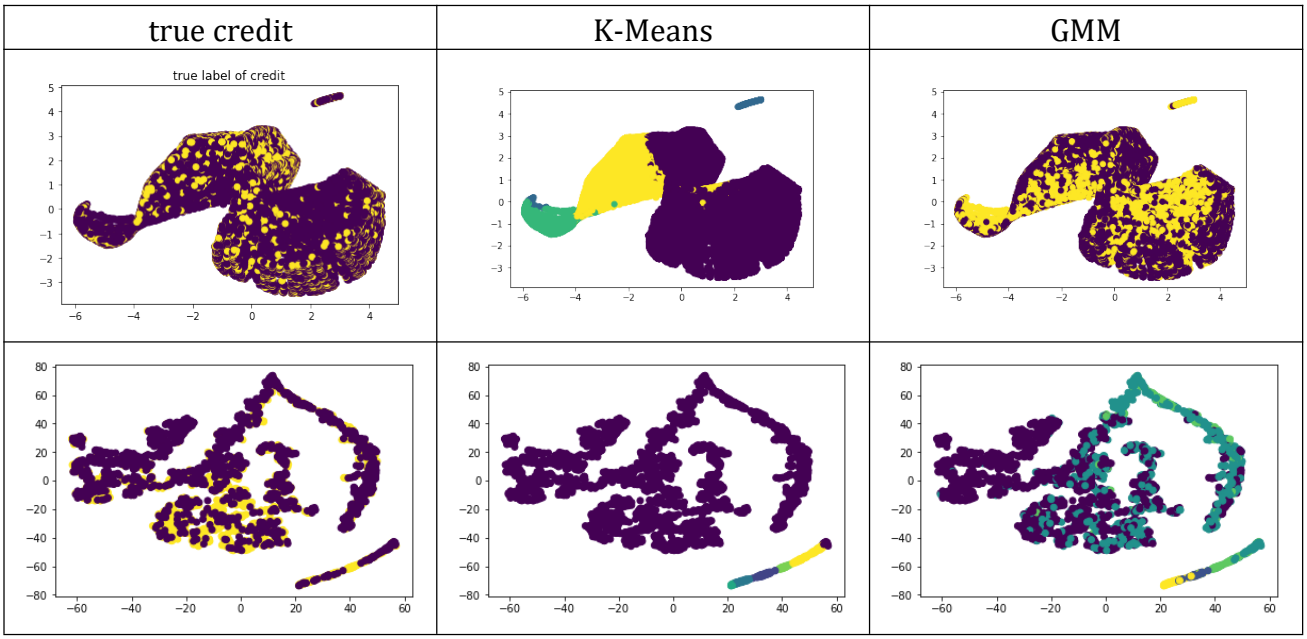


Fig. 3 Cluster visualization: Dataset 1(upper),Dataset 2(lower)

Take dataset 2 for instance, it seems that GMM cluster looks more like the true dataset, while in K-Means cluster most data points are put into one class(purple). The data points looks more like coupling with each other and GMM could find out the different (green) data points among them (purple). Due to page length, full results and analysis for both datasets can be found in the Jupyter Notebook.

3. Dimensionality Reduction and clustering results

i. Principle Component Analysis (PCA)

Eigenvalue in PCA reflects the importance of the corresponding component. As it is the coefficients attached to eigenvectors, they can measure data covariance and tell principal components in order of significance (Skymind, 2019) by ranking the eigenvalue from highest to lowest. I plot the eigenvalue (absolute value and ratio in left-below) of all principle components. The cumulative eigenvalue ratio is plotted in right-below. We can see that the first several principle components contain most information of both data sets.

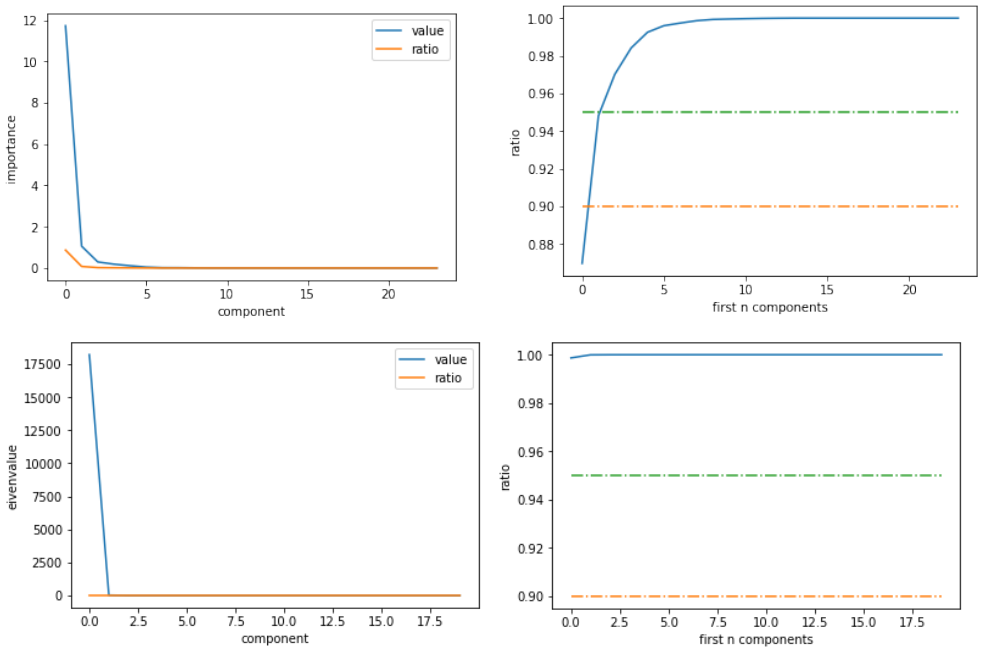


Fig. 4 Distribution of eigenvalues: Dataset 1 (Upper), Dateset 2 (Lower)

Then I reproduced clustering in dimensionality reduced data. Take dataset 1 credit data as example, the cumulative ratio printed are: [0.8697, 0.9482, 0.9702, 0.9841, 0.9925, 0.9960, ……], indicating PCA a good method for dimension reduction. I preserved theses 5 principal components as PCA feature for clustering. The similar procedures mentioned above was used to find the best k for K-Means with PCA features. Using the elbow method, again, I determined *k=3,* since is has relatively low SSE, and relatively high CH score and f1 score. As for GMM with PCA features, the ideal *k* is 3. For dataset 2, top 2 principal components contain over 99% information. Using similar process, *k=6* for K-means and *k=5* for GMM.

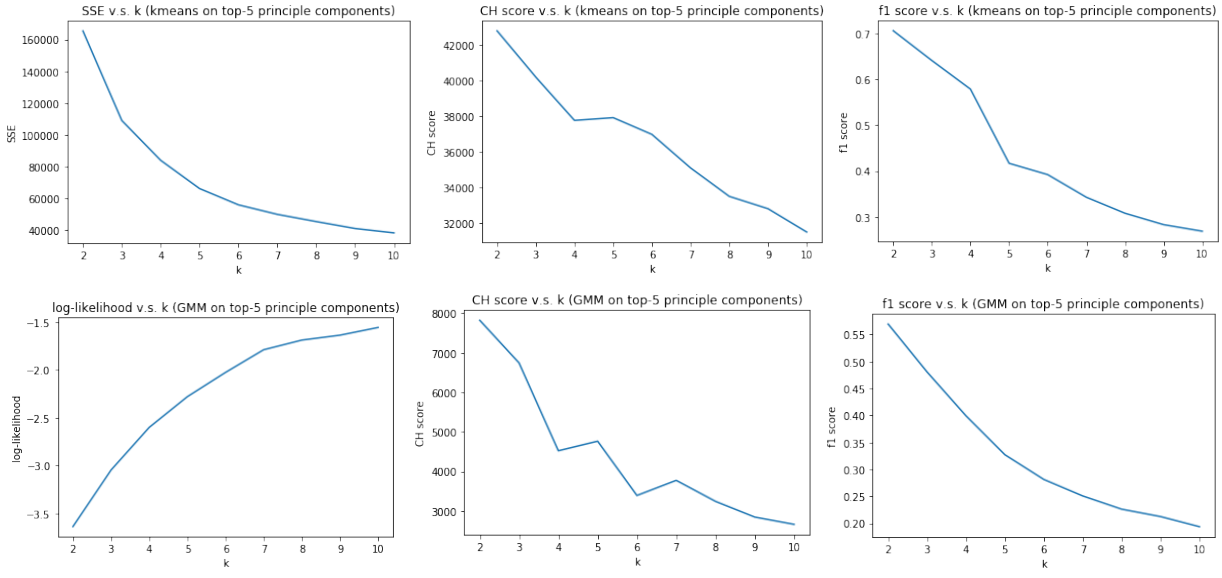


Fig. 5 Metric curves for K-cluster & GMM: Dataset 1

And the clustering results can be visualized by the first two features (as below).

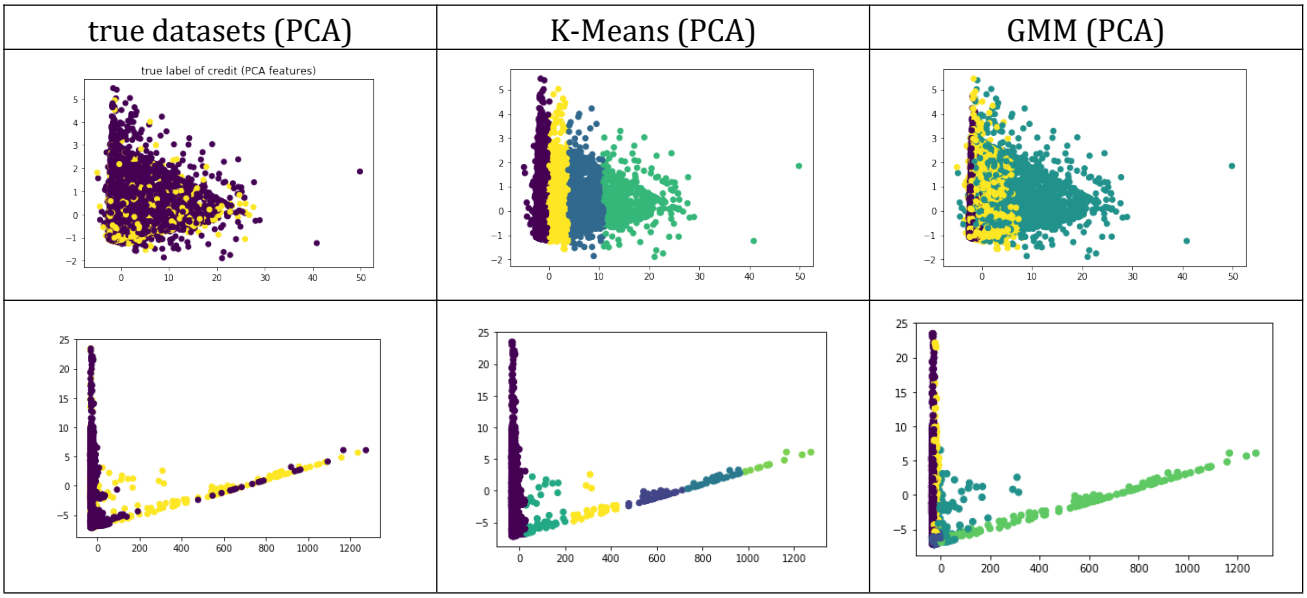


Fig. 6 Cluster visualization: Dataset 1 (Upper), Dateset 2 (Lower)

Take dataset 2 for example, it seems that K-Means clusters are more like the true gender on the PCA feature as K-Means clusters the data with x near zero as a class, however, GMM may assign wrong cluster to them.

ii. Independent Component Analysis (ICA)

ICA bases on non-Gaussian distribution assumption of samples, which can be evaluated via kurtosis (En.wikipedia.org, 2019). After performing ICA, I calculated kurtosis of all components. In order to check the independence degree of each component, I calculated and show the correlation of each feature after ICA. For dataset 1, the following figure shows that some features are independent from each other, while some are correlated. So, ICA and the non-Gaussian assumption may not suitable for it. ICA assumes that the source signals are independent of each other. However, in the case of dataset 1, several features considering default of credit card are apparently relevant and probably dependent. In this biased dataset, those tended to have payment delay are likely to delay again, leading to repayment status of a certain month relevant to its repayment status of previous month. For dataset 2, the result of ICA on clustering is better.

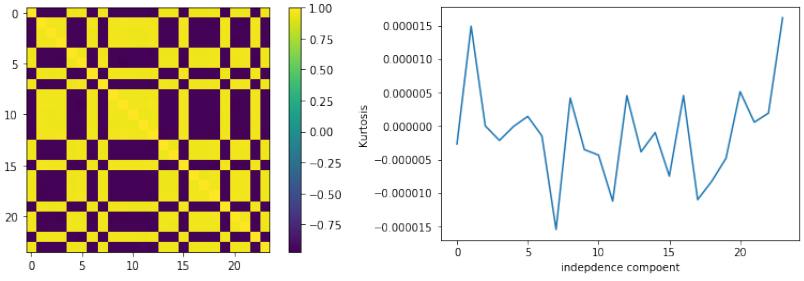


Fig. 7 Correlation of each feature after ICA (left); Kurtoic distribution (right): Dataset 1

I chose the number of components with highest kurtosis. The 5 features with top absolute Kurtosis are selected. Using the feature processed by ICA for clustering, again, I varied *k* to find the best number of cluster in K-Means. With the elbow method,I determined *k=5*. Similarly, the best *k* for GMM is 4. For dataset 2, two values are 3 and 6.

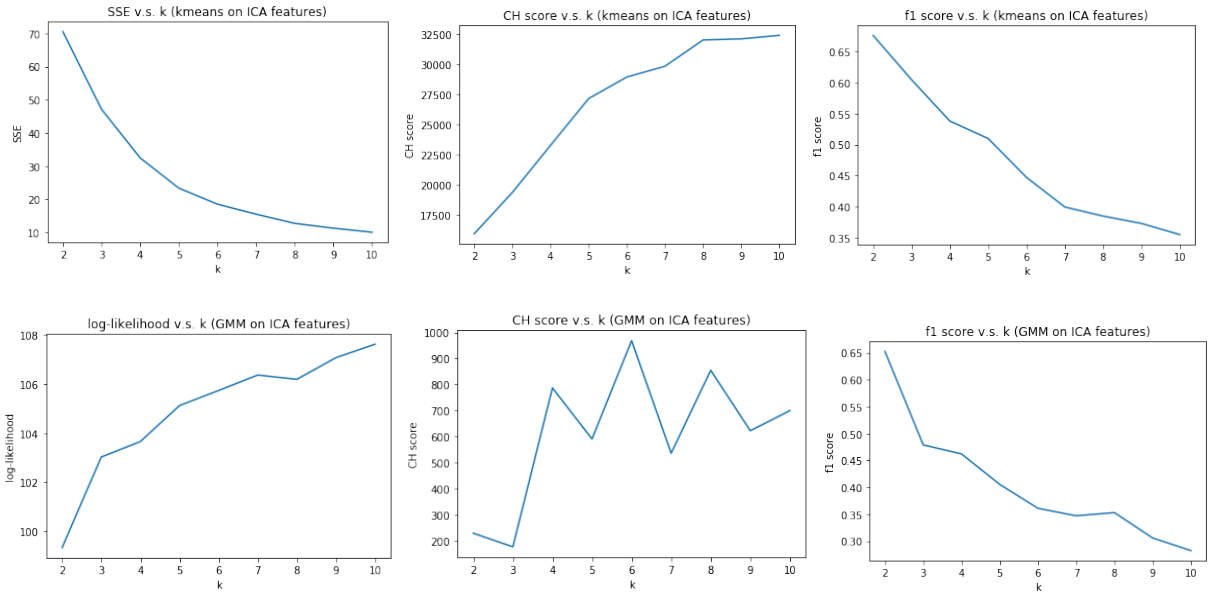


Fig. 8 Metric curves for K-cluster & EM : Dataset 1

And the clustering results can be visualized by the first two features (as below).

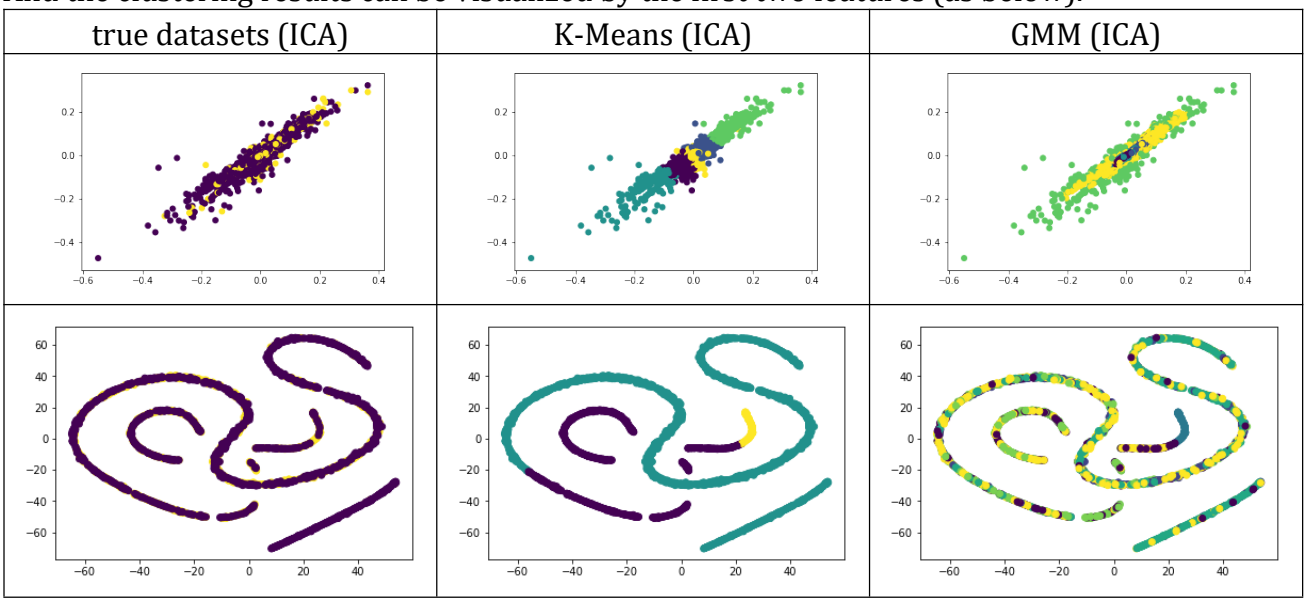


Fig. 9 Cluster visualization: Dataset 1 (Upper), Dateset 2 (Lower)

It seems that GMM clusters more like the true datasets on the ICA feature.

iii. Random Projection (RP)

Random projection transforms high dimensional data into low dimensional. To preserve more information, I calculated the reconstruction error (LDA, 2016) by minimal multiplier methods. After 5-fold cross validation, the number of components with least reconstruction error could be determined to be the optimal one (Fig 9). Thus, I chose 19 components for dataset 1 (with lower deviation) and 18 components for dataset 2.

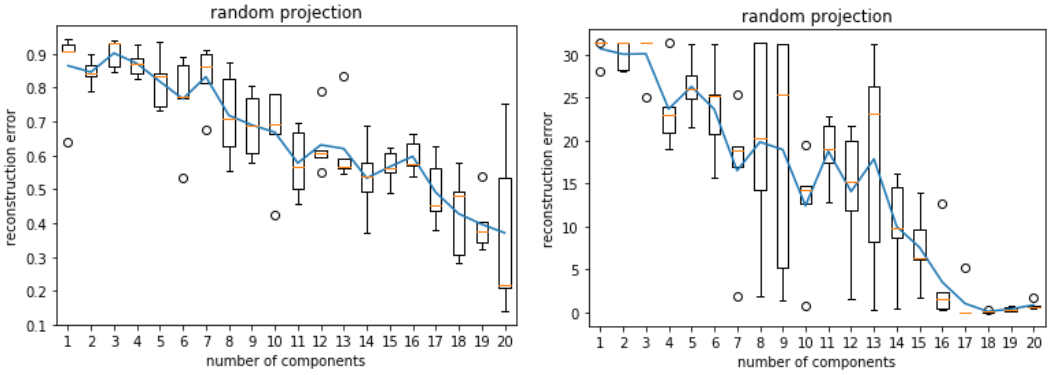


Fig. 10 Reconstruction Performance with Cross Validation: Dataset 1 (Left), Dateset 2 (Right)

Using the feature processed by RP for clustering, *k* is varied to find the best number of cluster in K-Means. With the elbow method, here, I determined *k=3* for K-means and GMM in dataset 1. For dataset 2, the two k values are 6 and 3.

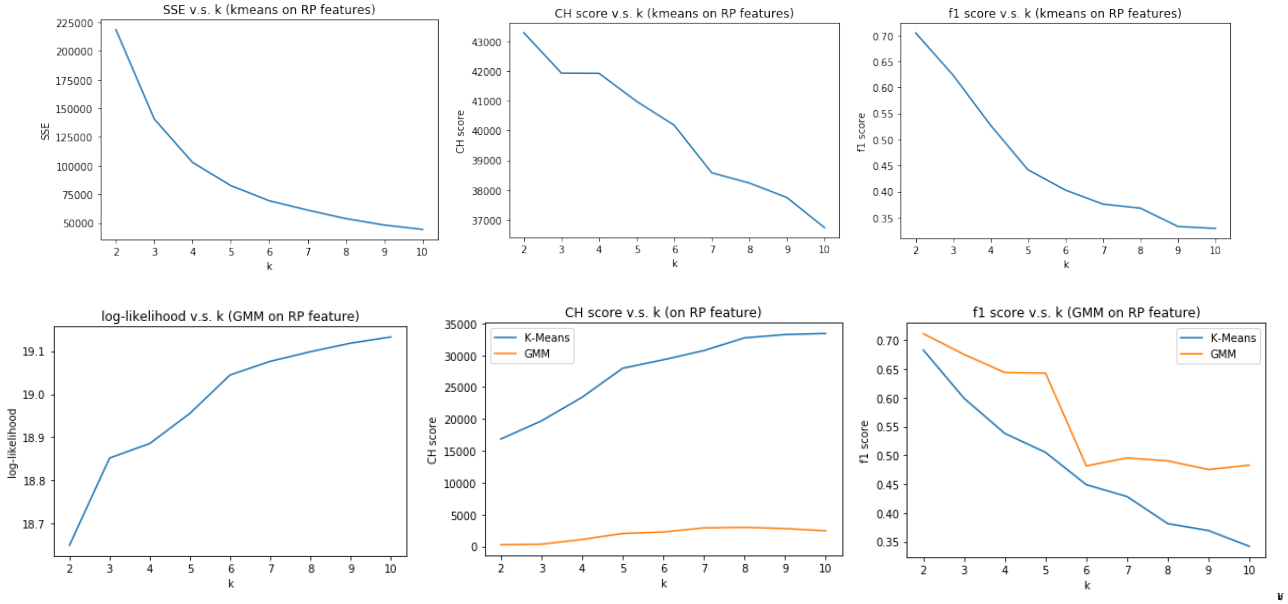
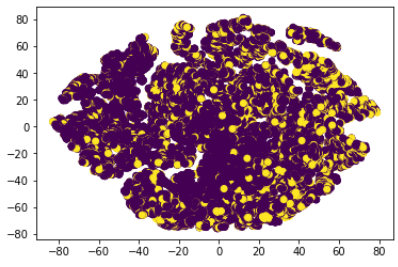
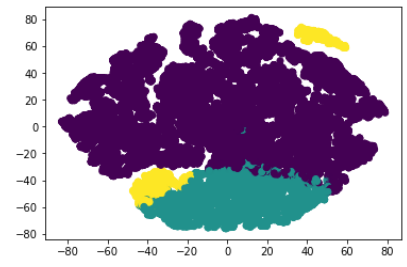
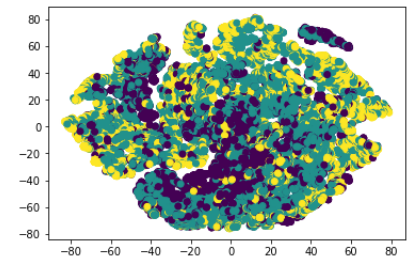


Fig. 11 Metric curves for K-cluster & EM using Random Projection : Dataset 1

As the result, the clustering results can be visualized below. It seems that GMM clusters are more similar to the true datasets after using RP dimensionality reduction.

true datasets (RP)	K-Means (RP)	GMM (RP)
		

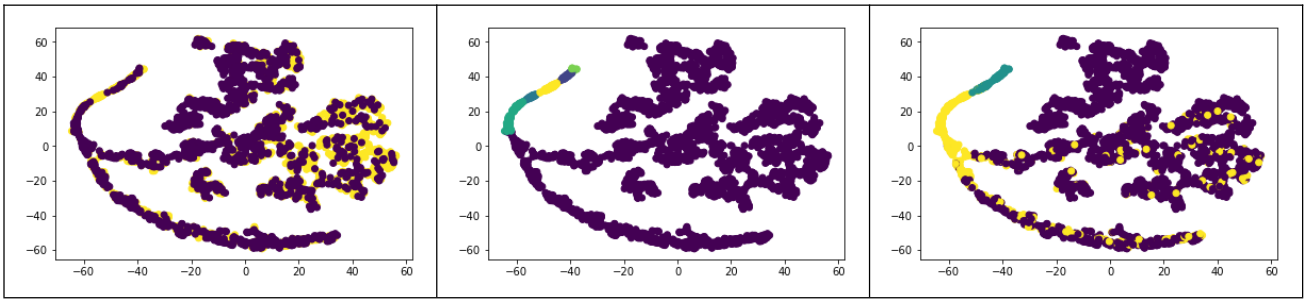


Fig. 12 Cluster visualization: Dataset 1 (Upper), Dateset 2 (Lower)

iv. Random Forest (RF)

To select features, random forest can be used as well. The feature importance are obtained when fitting labels (Medium, 2017).

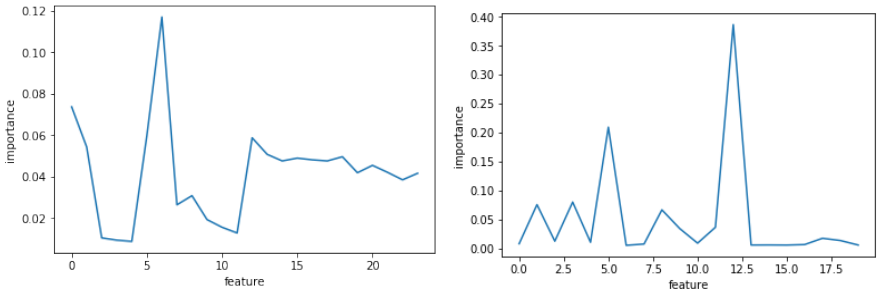


Fig. 13 Feature Importance using Random Forest: Dataset 1 (Left), Dateset 2 (Right)

Features with highest importance after processed by RF are chosen for clustering, *k* is varied to find the best number of clusters. I determine *k=3* for K-means and GMM in dataset 1. For dataset 2, the two k values are also 3.

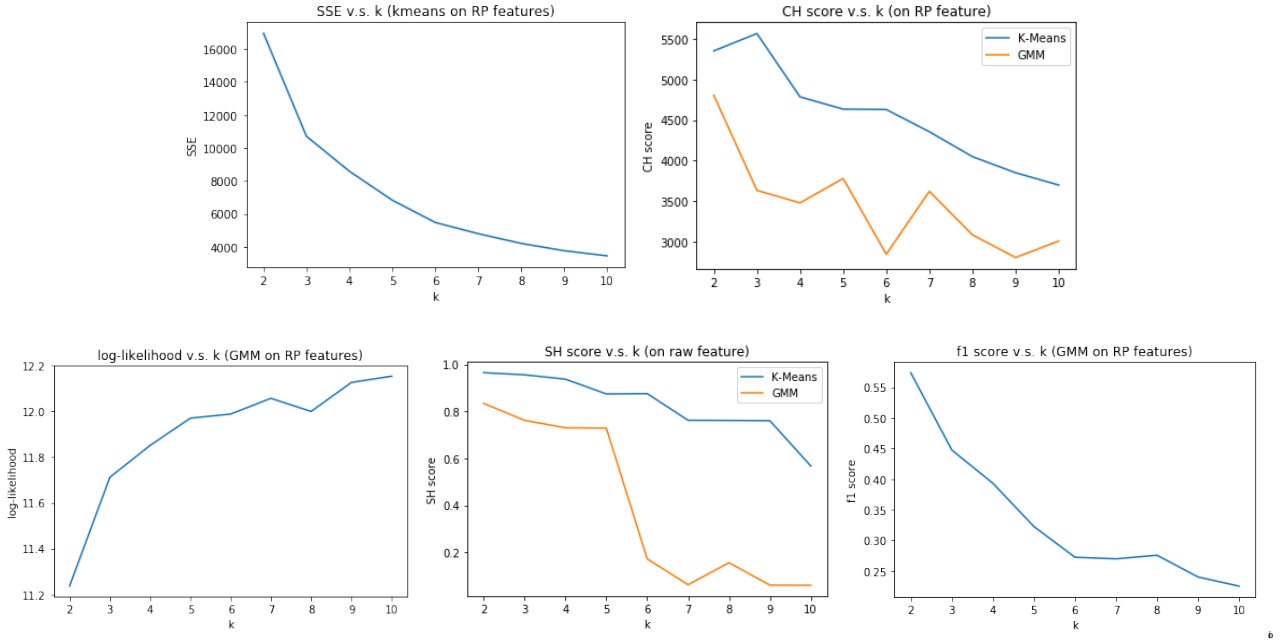


Fig. 14 Metric curves for K-cluster & EM using Random Forest: Dataset 2

And the clustering results can be visualized by the first two features. Due to its randomness, neither of the clustering in both datasets are very similar to the original one. For dataset 2, The performance of GMM and K-means are moderate.

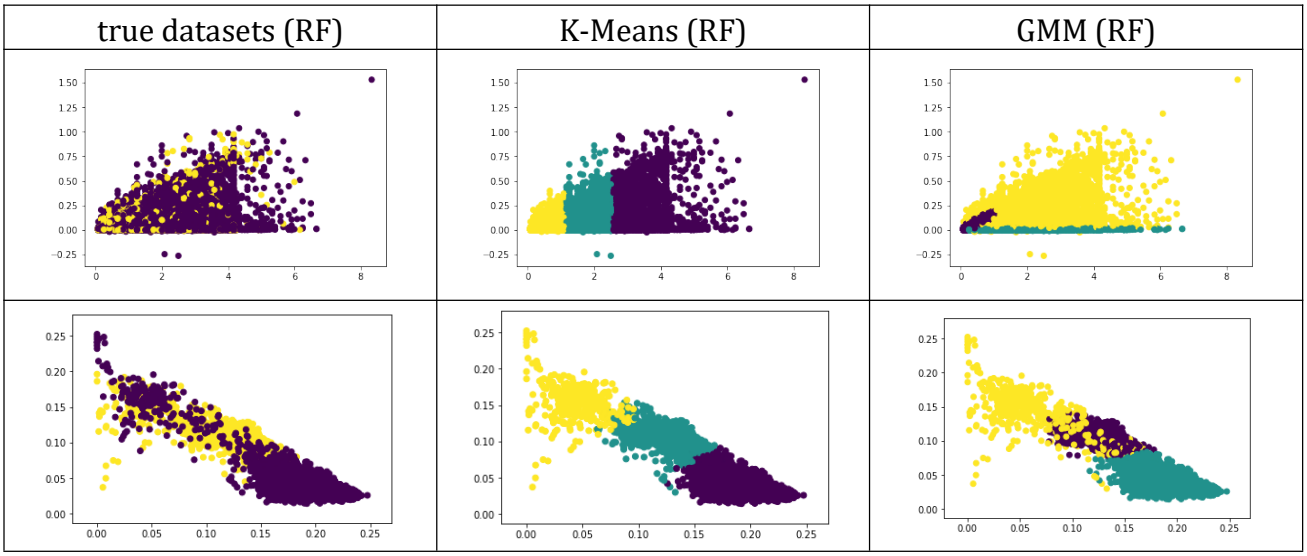


Fig. 15 Cluster visualization: Dataset 1 (Upper), Dateset 2 (Lower)

4. Using the clustering results to assist classification

The original Neural Network learner gives following results. I used a single hidden layer and studied how the performance will change as I use different number of neurons in this layer. For simplicity, I will only compare the results using 50 neurons, which is the optimal for original dataset with highest efficiency.

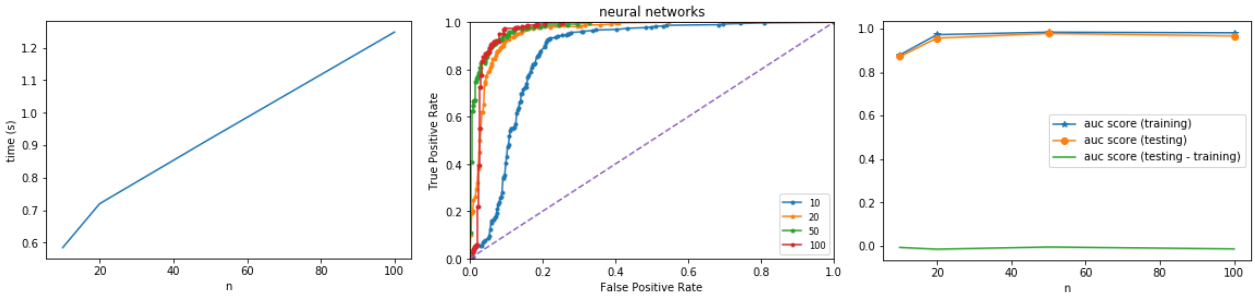


Fig. 16 Original Neural Network Analysis

1) Dimensionality reduction on dataset 2 gender voice

The acc, F1 and auc of test set are compared for n=50. It seems that PCA, ICA and RP do not increase the NN performance while random forest makes its contribution. But the result of RF is not always good. For time, the time cost is PCA<RP<ICA<RF.

2) Treating clustering algorithm as dimensionality reduction on dataset 2 gender voice which has applied dimensionality reduction.

Similarly, the acc, F1 and auc of test set are compared for n=50. Only the better clustering method with optimal components in each DR algorithm is chosen. It seems that neither algorithm increase the NN performance. Anyway, except RP, other ones are acceptable. If they are combined together, the accuracy and auc score is higher than any of them. For time, the time cost is RF<RP<ICA≈PCA.

Algorithm	acc	F1	auc
N/A	0.9217	0.9229	0.9780
PCA	0.7800	0.6829	0.7046
ICA	0.6629	0.7010	0.7391
RP	0.7449	0.7554	0.8280
RF	0.9722	0.9726	0.9905

Algorithm	Cluster	acc	F1	auc
N/A	EM	0.9331	0.9332	0.9799
PCA	KMean	0.9154	0.9198	0.9705
ICA	EM	0.9104	0.9077	0.7391
RP	EM	0.8876	0.8932	0.9499
RF	KMean	0.9318	0.9332	0.9663

Fig. 17 Neural Network Comparison: DR on original dataset 2 (left); Clustering on DR applied dataset 2 (right)

5. Conclusion

1) RP is affected by randomness, which can be seen from the cross validation of reconstruction error. RF has similar issue. In contrast, PCA and ICA are relatively stable

2) For my datasets, EM is often better than KMeans. The clusters of KMeans are often near in space while EM performs better finding points of different categories although they are near.

3) Dimension reduction+NN is basically not as good as the original one. Probably this is because it loses discriminating information. NN itself has a strong ability to capture features. With dimension reduced, the important features become harder to find.

4) Cluster+NN performs somehow better. It helps classify the samples and improve the performance of NN learner.

All in all, feature reduction algorithms are mostly used on datasets with extremely high dimension (over 100 features). In those cases, the feature reduction effect are more significant. But for my datasets with 20-30 features, they are not that effective.

References

1. Kaggle.com. (2019). *Default of Credit Card Clients Dataset.* [online] Available at: https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset
2. Kaggle.com. (2019). *Gender Recognition by Voice.* [online] Available at: https://www.kaggle.com/primaryobjects/voicegender
3. Scikit-learn.org. (2019). *scikit-learn: machine learning in Python* — scikit-learn 0.16.1 documentation. [online] Available at: https://scikit-learn.org/
4. Scikit-learn.org. (2019). *sklearn.metrics.calinski_harabasz_score — scikit-learn 0.21.3 documentation*. [online] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.calinski_harabasz_score.html.
5. Skymind. (2019). *A Beginner's Guide to Eigenvectors, Eigenvalues, PCA, Covariance and Entropy*. [online] Available at: https://skymind.ai/wiki/eigenvector
6. LDA, M. (2016). *Meaning of "reconstruction error" in PCA and LDA*. [online] Cross Validated. Available at:https://stats.stackexchange.com/questions/194278/meaning-of-reconstruction-error-in-pca-and-lda [Accessed 4 Nov. 2019].
7. Medium. (2017). *Machine Learning: Unsupervised Learning — Feature selection*. [online] Available at: https://medium.com/machine-learning-bites/machine-learning-unsupervised-learning-feature-selection-a9bdccd70f95
8. En.wikipedia.org. (n.d.). *T-distributed stochastic neighbor embedding*. [online] Available at: https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding
9. Taboga, M. (2019). *Log-likelihood*. [online] Statlect.com. Available at: https://www.statlect.com/glossary/log-likelihood.