Clustering and Dimensionality reduction Algorithms

Surabhi Amit Chembra
surabhichembra@gatech.edu

Abstract:
This paper explores unsupervised clustering algorithms of k-means and Expectation maximization, dimensionality reduction algorithms of principal component analysis (PCA), Independent component analysis (ICA), Random Projection (RP), Random Forest (RF) by using two datasets: letter recognition and bio response. Furthermore, these techniques are used to pre-process the data fed into the neural network built in Assignment-1(A1) and the performances are compared.

Dataset description: Both datasets are classification problems downloaded from openml.org website. The letter recognition dataset[1] from Assignment-1 has 16 attributes, each scaled from 0 to 15 and 20000 instances. Each row represents a capital letter (A to Z). The dataset is balanced with around 750 instances per letter, as in Fig.1. The second dataset chosen is "Bioresponse" with 3751 instances, 1777 features ranging from 0 to 1 each and 2 classes. The attributes are the calculated properties of molecules and the target class says whether the molecule exhibited actual biological response (1) or not (0) in the experiment. The dataset is slightly unbalanced with 2034 instances for class 1 and 1717 with class 0. Fig.1 shows class distribution for the attributes. Since there are many dimensions, I have shown only first 20 dimensions in Fig.1.b due to spatial reasons.

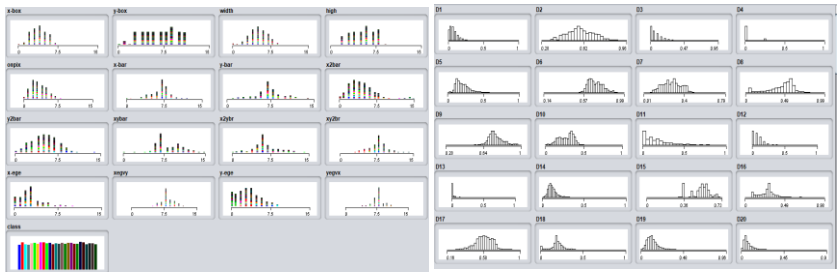


Fig.1: Class distribution for attributes in Letter Recognition[left] and Bioresponse[right]

Why are the datasets interesting?
From application point of view: Recognizing handwritten letters helps to decipher handwritten notes and store the content electronically online, thus preserving knowledge even when paper-notes are lost. Lesser the paper-notes, lesser the trees cut to make those. Also, AI agents capable of understanding handwritten notes can interact better with humans. Regarding Bioresponse dataset, being able to predict if a molecule will respond, based on its calculated properties reduces the number of actual biological experiments needed and quicken up the testing of new medicinal drugs.
From Machine Learning point of view: Initially, I tried small datasets like Iris (150 instances, 4 features, 3 classes) and found that the clustering does well as expected. But since the features are less, this was not a suitable candidate for exploring dimensionality-reduction (DR). So, to understand how Clustering and DR behaves in large feature-spaces, I chose Bioresponse dataset with **1777 features** and 3751 instances. In real world, DR is mostly needed/used for such datasets to combat curse of dimensionality caused by many features and less instances, and it was interesting to explore how well DR works for such cases. However, both the above datasets have a smaller number of classes and I was interested to see how clustering and DR works when the number of classes is high, and letter recognition dataset from A1 was suitable for that, with its **26 classes**. It was intriguing to explore what effect does preprocessing of input data (letter recognition dataset) by clustering and DR has on the performance of the Neural Network from A1.

Experimental Methodology :This assignment is coded using Python 3.6 and Windows 10 machine. Scikit-learn was used to implement algorithms. MS Excel and matplotlib were used to plot graphs. Part-1 focuses on clustering for both datasets, while Part-2 analyses how preprocessing by DR affects clustering and Part-3 focuses on Neural-Network analysis. Since in Assignment-1, 30% of letter recognition dataset was kept aside for testing, the same pattern is continued here and the same neural network from A1 was used, to maintain consistency for Neural Network performance.

Part 1: Clustering
1. k-means: K-means clustering is implemented using sklearn.cluster.KMeans().The algorithm picks k centers and each center claims its closest points. Then, the centers are recomputed by averaging the clustered points and this process is repeated until convergence. Thus, k-means is effectively an optimization algorithm that gives compact clusters without giant holes in them. k-means is a fast and simple algorithm, it falls in local minima and thus is

recommended to restart it several times and to spread out initial cluster centers. The objective of k-means is to minimize within-cluster variance which is same as sum of squared Euclidean distances of data points from their cluster centers. The convergence proof assumes that the cluster assignment step and mean update step optimize the same criterion of minimizing within-cluster variance. As there are a finite number of assignments, convergence is guaranteed after finite steps. It may stop converging with other distance measures unless another appropriate center estimation method is chosen. Also, k-means repeatedly assigns data points to the closest centroid, which is multivariate mean in Euclidean space. Since Euclidean space is mainly about Euclidean distances, it is used as the measure of distance/similarity. Regarding time complexity, each iteration is of the order O(kn) and there are maximum $O(k^n)$ configurations (one for each iteration). As per "no free lunch" theorem, there is no optimal K suitable for all scenarios and algorithms. Since there are no principled statistical methods that can set right k in all cases, heuristics or rules of thumb like elbow method is used. Since the K-Means score is the negative of the objective function of K-Means, which is to reduce the sum of squares of the intra-cluster distances or within-cluster sum of squared errors (SSE), the lesser the score, the better. The negative of score value tells how internally coherent the clusters are. Hence, I used negative score value to determine optimal k. In Fig2.a, the elbow is at 40. For letter dataset, past 40 clusters, there is no commendable decrease in SSE. However, minimum SSE will be when each data point is its own cluster, which is inefficient clustering. So, to incorporate the between-cluster sum of squared distances, I used silhouette score to validate the chosen k. Silhouette method validates the consistency within clusters of data. The silhouette of an instance tells how closely it is matched to data within its cluster and how loosely it is matched to data of the other neighboring clusters. A value near 1 implies that the instance is in the apt cluster whereas a value near -1 implies that the instance is an outlier, that is, it is in the wrong cluster. In the Fig.2.b for letter recognition, many instances have value more than 0.2 when 40 clusters are used, as compared to 10 clusters in Fig.2.c (both 10 and 40 are shown for comparison). The Silhouette is calculated here based on Euclidean distance. Since the dataset is balanced, equal-sized clusters are preferred, and most similar sized clusters were obtained with k=40. The Bioresponse dataset has its elbow at 4 in Fig.3.a and the silhouette plot in Fig.3.b shows that cluster0 has many data points with more than 0.2 value, though the other clusters have more outliers (negative value). Also, as the dataset has 45% (class 0)-55% (class 1) class separation, cluster 0 might be representing the 55% portion, though it is bigger than 55%. When number of clusters was chosen as 10 in Fig3.c, only cluster 0, 5 and 9 had positive values and others had negative value, thus showing less intra-cluster consistency. So, 4 is chosen as the cluster number.

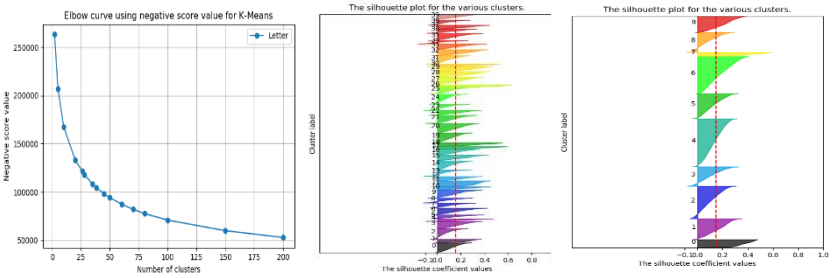


Fig.2. Elbow curve [Fig 2.a], silhouette plots [Fig 2.b and 2.c] for letter recognition

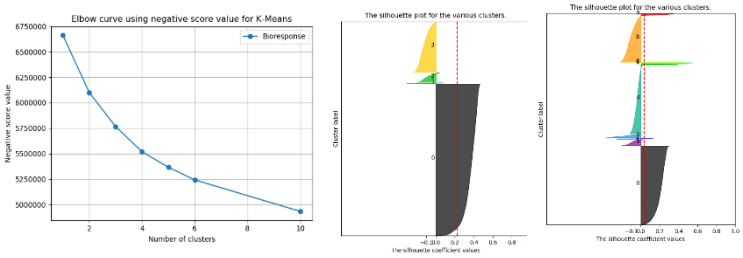


Fig.3. Elbow curve[Fig 3.a] and silhouette plots [Fig 3.b and 3.c] for

2.Expectation maximization: EM is implemented using sklearn.mixture.GaussianMixture(). This soft-clustering method assumes that the data was generated by k gaussians with fixed known variance and finds a hypothesis or set of gaussians that maximizes the probability of generating the given data. Each data point can belong to multiple clusters probabilistically. EM acknowledges the fact that gaussian has infinite extent, so any point belonging to a cluster with probability 0.9995 can still belong to any distant cluster with non-zero but very small probability. The computed likelihood is monotonically non-decreasing and thus get better in each iteration, though not guaranteed to converge due to infinite

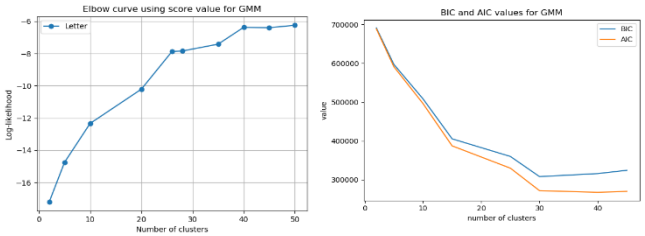


Fig.4. Likelihood[Fig 4.a] and BIC[Fig 4.b] plots for letter recognition

configurations possible in the probability space. The weights, means and precisions can be initialized using k-means (set init_params as 'kmeans') or randomly(set init_params as 'random'). The **score metric for gaussian mixture model** (GMM) computes the logarithm of the probability(likelihood) of clustering. Since the probabilities are in the [0,1] interval, the score metric is negative in value, as in Fig.4.a. Since increasing the number of clusters or parameters can improve the likelihood, it can cause overfitting. So, Bayesian Information Criterion (BIC) and Akaike Information criterion(AIC) introduces a penalty term for the number of parameters to combat overfitting and the lowest BIC or AIC value is preferred. The penalty term is larger in BIC than in AIC. Since the BIC and AIC values are the least at 30 in Fig.4.b, I chose 30 as optimal k. The covariance-type for the GMM model can be full, where each component has its own general covariance matrix, tied where all share the matrix, or diagonal where each has its own diagonal matrix or spherical with each having its own single variance. It also decides how the clusters will be aligned in the space. The BIC score for different types is shown in Fig.5. It can be observed that full(orange) has lowest BIC and thus full type was used here. For Bioresponse dataset, AIC is lowest at 5 clusters and BIC at 2 clusters in Fig.6.b. The log-likelihood curve improves a lot from 2 to 5 clusters but does not show commendable improvement past 5 clusters in Fig.6.a. So, I chose 5 as optimal k.

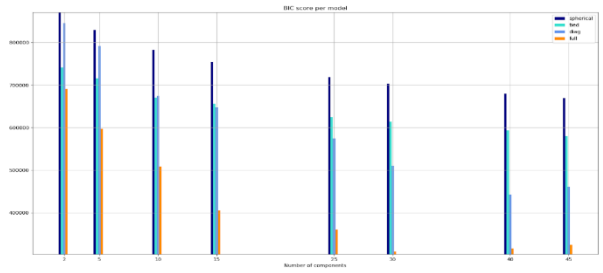


Fig.5. BIC plots for different covariance types

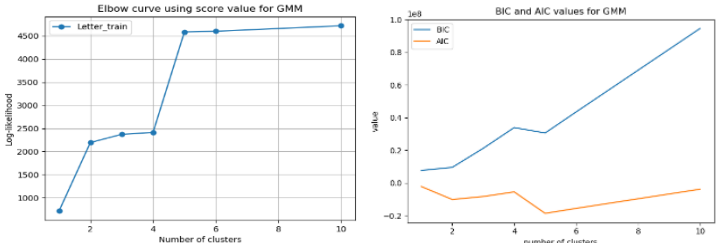


Fig.6.Likelihood[Fig.6.a] and BIC[Fig.6.b] plots for bioresponse

Part 2: Dimensionality Reduction

PCA: Principal Component Analysis is a linear feature transformation algorithm that finds the mutually orthogonal components with maximum variance for the data points. This global constraint of mutual orthogonality reduces reconstruction error. The dimensions with maximum eigen values are considered, as more eigenvalue implies more contained information and thus more important feature. A component with 0 eigenvalue is ignorable as it has no variance. PCA is implemented using sklearn. decomposition. PCA(). The eigenvalues of all features of letter recognition are plotted in Fig.7.a and Fig.7.b shows the percentage of variance explained by each component. I chose 10 components because past 10, the eigenvalues are very less, 10 components together have explained variance of 91% and the reconstruction error for 10 components is less than 0.1 (Fig.7.c). Fig.8 shows both datasets in new PCA data spaces and the t-SNE plots in 3-D for the PCA applied data. It is noticeable that the dimensions are mostly gaussian and in t-SNE plot for bioresponse, the front part is cluster1(yellow) and back portion mostly cluster0. The huge number of classes in letter recognition makes cluster separation tough to see in t-SNE plot (also multiple clusters are assigned same color by default as there are many clusters). Red, green, blue and yellow clusters are perceivable

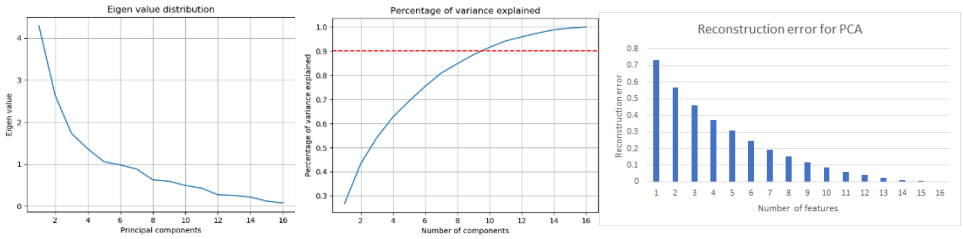


Fig.7.Eigen value[Fig.7.a], variance explained[Fig.7.b] and reconstruction error[Fig.7.c]

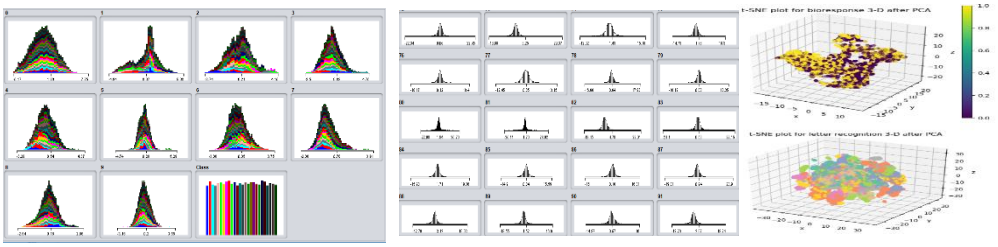


Fig.8.Data in new spaces and t-SNE for letter[Fig 8.a, 8.d] and bioreponse [Fig 8.b, 8.c]

though.The PCA applied data is then used for clustering. The elbow is spotted at 20 in Fig.9.a. and the silhouette plot for 20 clusters shows roughly similar sized

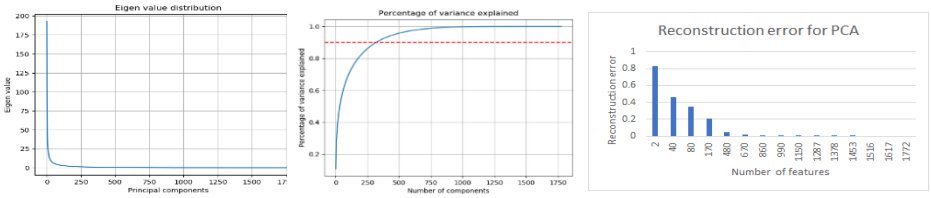


Fig.10.Eigen value and BIC plots for bioresponse

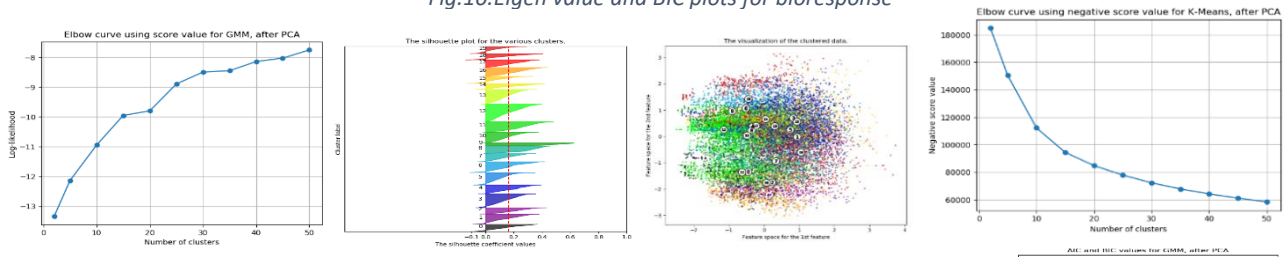


Fig.9.Finding optimal k for letter recognition

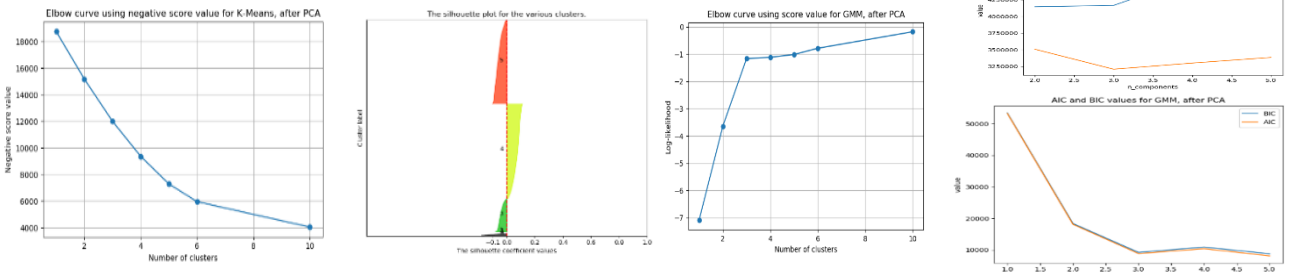


Fig..11. Finding optimal k for Bioresponse

clusters and many data points with score more than 0.2. Fig 9.c shows the clustering in 2-D space. Though the clusters are not well-separated, they are better

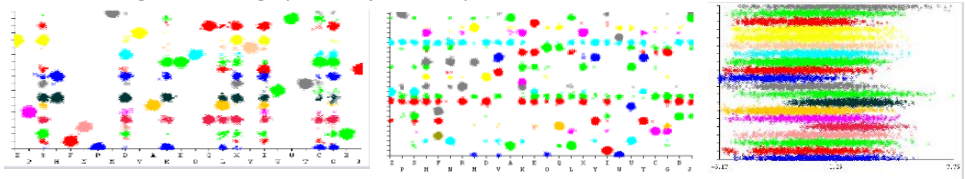


Fig.12.Class-cluster alignment for letter recognition

identifiable than before applying PCA. If represented in 10-D space (PCA gave 10 features), separate clusters will be more visible. For GMM, 35 clusters were chosen as Fig.9 gives lowest BIC and AIC values at 35 clusters and past 35, there is no significant improvement in log-likelihood. For Bioresponse dataset, in Fig.10, 319 components have explained variance of 90% and past 319, the eigen values are not significant and the reconstruction error is less than 0.05. In Fig.11, the elbow curve shows no significant improvement past 4 clusters. and the silhouette plot for 4 clusters shown in Fig.11.b has cluster 2 and 3 similar-sized. For GMM, BIC and AIC give least value at 3 and no significant improvement is spotted in likelihood past 3 clusters. So, 3 clusters are chosen for GMM. Fig.12.a and 12.b show how the clusters line up with classes for k-means and GMM respectively. X-axis represents 26 classes and Y axis the clusters. For example, in Fig.12.a, cluster1(blue) consist of mainly letter P(6th class) and in Fig.12.b, cluster 0 (blue) consist of letter W(20th letter). Fig.12.c shows the features in 1-D space and Fig.13 shows clustering in 2-D space for k-means and GMM. Fig.14.and14.b show the class-cluster alignment for K-means and GMM respectively for bioresponse where X axis represents cluster and Y-axis class. K-means did a good job of separating clusters such that cluster3 has class 0 instances, whereas all the other three clusters have

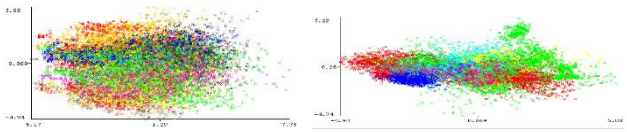


Fig.13.Clusterings produced for letter recognition

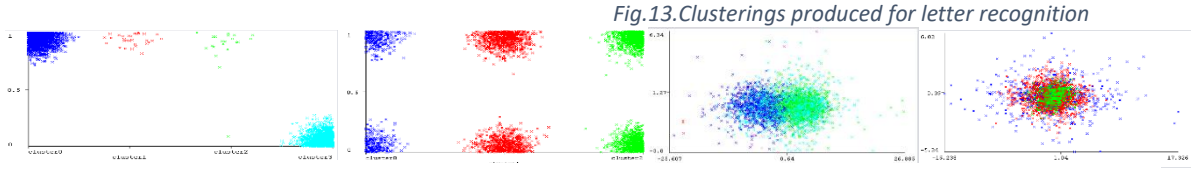
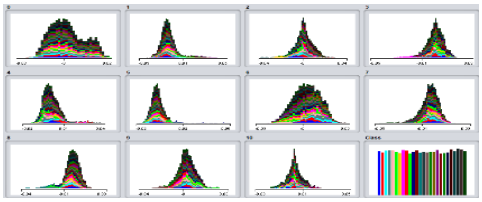
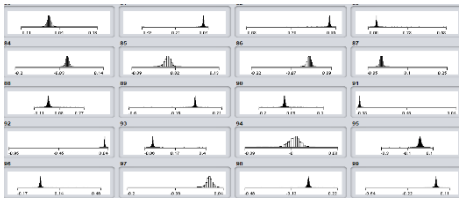


Fig.14.Class-cluster alignment [LEFT] and clustering formed [RIGHT] for bioresponse

class 1 instances. GMM did not cluster properly because all three clusters have instances from both the classes. and Fig.14.c and 14.d show the clustering result of k-means and GMM respectively in 2-D. There are two big clusters in k-means which has 53% and 46% of instances, and the rest of the instances are outliers captured by other 2 small clusters. GMM has similarly-sized 3 clusters.

2.Independent Component Analysis (ICA): ICA separates signals from different sources assuming that the source signals are independent of each other and that the values in each have non-Gaussian distributions. The independent components with non-Gaussian histograms or with less complexity are chosen as the new dimensions by ICA. It maximizes independence and all new features are statistically independent of each other. ICA finds the independent components such that the mutual information between them is minimized or their non-Gaussianity is maximized, which explains why the features are not so gaussian in Fig 15.a and 15.b. The minimization of mutual information uses Kullback-Leibler Divergence and maximum entropy, whereas non-Gaussianity uses kurtosis and negentropy based on central limit theorem. Kurtosis is the fourth central moment divided by the square of the variance. If kurtosis is higher than 3 or "excess kurtosis" is higher than 0, then the dataset has heavier tails than a normal gaussian distribution. If kurtosis is less than 3, it means the distribution has a flatter tail than a normal one with same mean and standard deviation. In this assignment, ICA is implemented using sklearn.decomposition.FastICA() and I used pandas dataframe to measure kurtosis which returns an unbiased kurtosis using Fisher's definition of kurtosis where 3.0 is subtracted from result to give 0 kurtosis for Gaussian distribution (excess kurtosis).The excess kurtosis for each ICA instance are shown in Fig.16.a. The Y axis values are cumulative kurtosis, that is, IC8 represents the kurtosis of 8 IC components chosen by FastICA algorithm. Even IC2 gives non-zero kurtosis, however, to maximize the non-Gaussianity without using all the components, I choose 11 components because past 11, the improvement in non-gaussianity is not so significant. After application of ICA, clustering is performed. Fig.15 shows the t-SNE plots for letter recognition with 26 classes and bioresponse with 2 classes after ICA. The bioresponse dataset has the blue cluster with class0 instances and yellow cluster with class1 instances. For k-means, in Fig.16.b, 20 clusters are given by the elbow method and the silhouette plot for 20 clusters do not show much outliers (almost all data instances have positive value). For GMM, BIC gives least value at 35. For Bioresponse dataset, even when only 2 dimensions are used for ICA, the excess kurtosis is greater than 0, hence non-gaussian. Using all components give a high non-Gaussianity. It can be noticed that using 1750 components has less kurtosis than 1500 components. So, discarding some dimensions actually increased cumulative kurtosis or non-gaussianity and those must be Gaussian features.. Though 1500 gave highest kurtosis, the huge number of components can cause curse of dimensionality. So, 302 was chosen among them as cumulative excess kurtosis of 302 is reasonable in Fig.17.a. The ICA applied data was re-clustered using k-means and GMM. The elbow curve in Fig.17.a gives 5 clusters for kmeans. BIC and AIC curves give least value at 6 clusters and the likelihood is high at 6 clusters, for GMM. Fig18.a and b show the cluster-class alignments for k-means and GMM respectively, with class on X axis, for letter dataset. It can be observed that most clusters own instances of one class in both cases, thus the clusters have more homogeneity than in PCA. For example, cluster0 in k-means owns mostly R instances and cluster1 in GMM has mostly M instances.





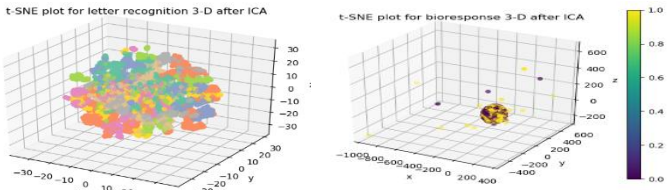
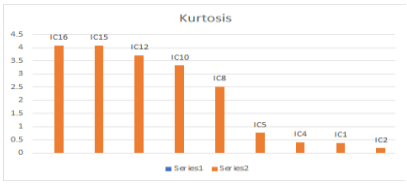


Fig.15.Data in new spaces [TOP] and t-SNE plots [BOTTOM] for letter [LEFT] and bioresponse [RIGHT]



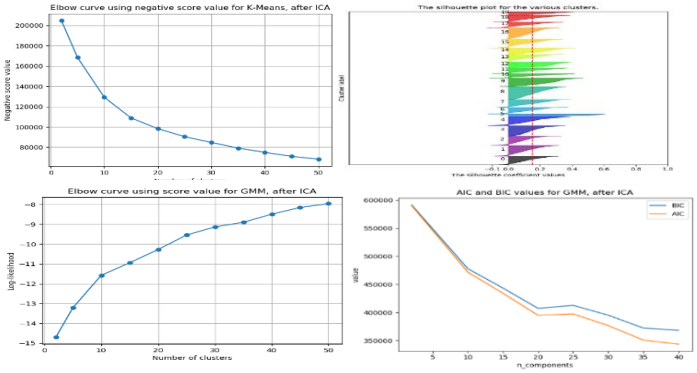


Fig.16.Kurtosis and finding optimal k for Letter recognition

For Bioresponse dataset, Fig.19. c and Fig.19.d show the clusters formed using k-means and GMM respectively. In k-means, the three major clusters account for most of the instances, with green and blue clusters owning most of class1 instances and the beige one owning most of class0 instances. As with PCA, k-means clusters have good internal consistency. GMM clustering is not much aligned with class labels and all 4 clusters have instances of both classes. So, GMM did not do well for this dataset, unlike k-means, presumably because GMM does well with gaussian datasets and this one is far from Gaussian (some features have value 0 or 1). Also ICA choose non-Gaussian features which puts GMM at a disadvantage.

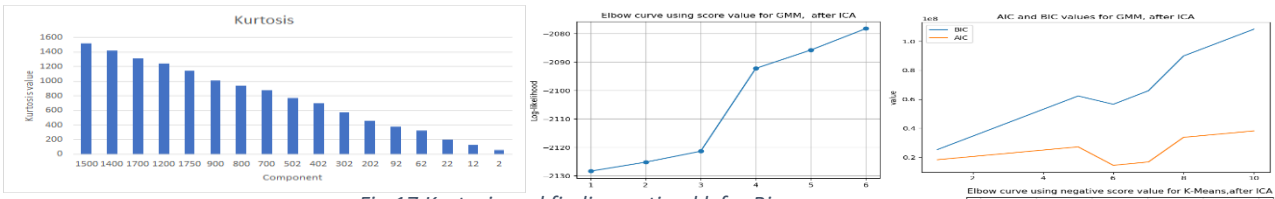


Fig.17.Kurtosis and finding optimal k for Bioresponse

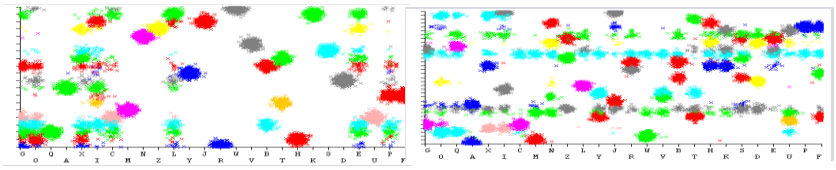


Fig.18.Class-cluster alignment for letter

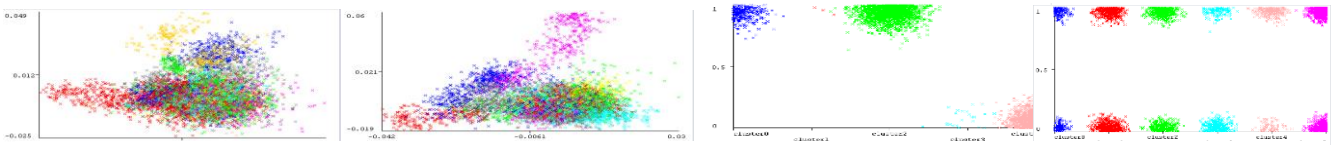


Fig.19. Class-cluster alignment [RIGHT] and clustering formed [LEFT] for bioresponse

Random Projection (RP): RP is implemented using sklearn.random_projection(). PCA projects high dimensional data along the components or axes with maximum variance. RP also projects high dimensional data to lower dimensions but on random components or axes. This explains why PCA (Fig.7.c) had less reconstruction error than RP (Fig.21.a). For letter recognition, if 10 components are selected, PCA gave 0.08 as error while RP gave 0.37, but RP is faster than PCA. According to the famous Johnson Lindenstrauss lemma, 8488 is the required number of dimensions for the letter recognition dataset of 20 K samples if the tolerance is 0.1 epsilon. However, since there are only 16 features,

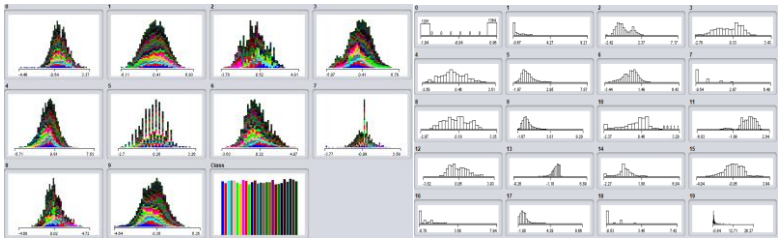


Fig.20.Data in RP spaces for letter [RIGHT] and bioresponse[LEFT]

Fig.21.Reconstruction error

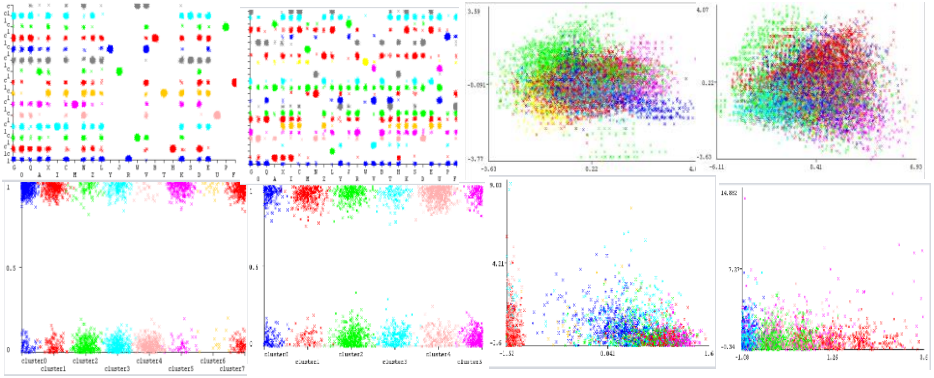


Fig.24.Class-cluster alignment[LEFT] and clustering formed [RIGHT] for letter[TOP] and

reconstruction error and accuracy are used to decide the number of features. Allowing for a maximum reconstruction error of 0.38, I chose 8 components as in Fig.21.a. Fig 20.a and 20.b shows the data in new spaces, for letter recognition and bioresponse respectively. It can be noticed that the features are not strictly gaussian as in PCA, they are just random components. Due to this randomness, for each seed set, the performance and reconstruction error can vary. Hence I ran for 10 different seeds and plotted the data for 4 of those with good variations in values. Seed 5 comparatively performed bad, while seed 8 showed good performance in Fig.21.a. RP-applied dataset was then used for clustering. The elbow curve in Fig.22.a. gives 15 clusters for k-means and the silhouette plot for 15 clusters shows most of the data points having positive score, with many scoring more than 0.2. In Fig.22, BIC and AIC values are least at 25 clusters and the likelihood does not rise much after 25 clusters, so I chose 25 clusters for GMM.

For Bioresponse dataset, Fig.21.b. gives 1378 components if the maximum allowed reconstruction error is 0.22. Regarding clustering, though a clear elbow is not present, SSE forms an elbow at 8 in Fig.22.b and has no much significant changes thereafter. The silhouette plot for 8 clusters shown in Fig.22.c has a huge blue cluster and two medium sized clusters, and the overall score is not bad, as the blue cluster has good positive score. So, 8 clusters were chosen for k-Means. Since both BIC and AIC values are least at 6 clusters and the log-likelihood graph does not give a clear pattern, possibly due to the projection onto randomly chosen axes, I chose 6 as the cluster number for GMM.

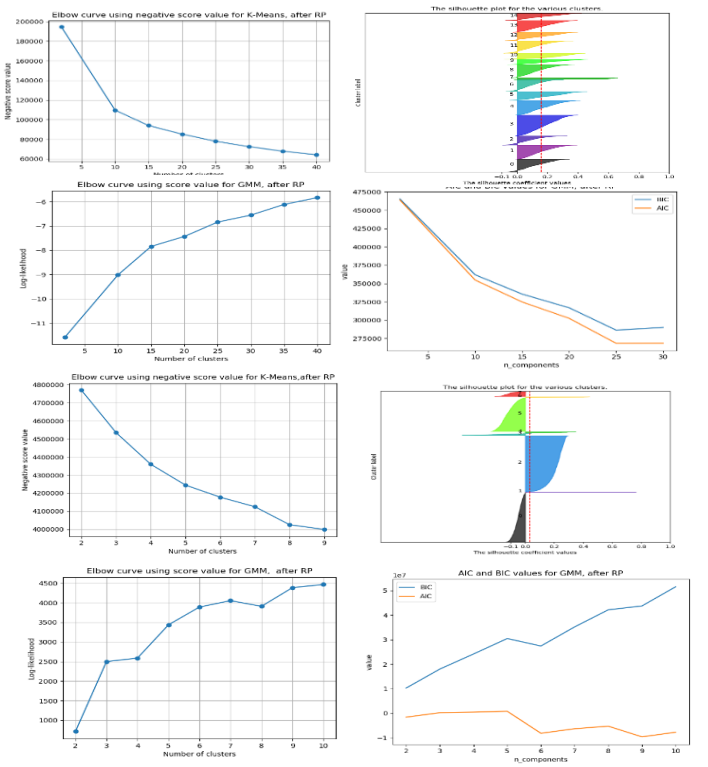


Fig.22. finding optimal k for letter and bioresponse

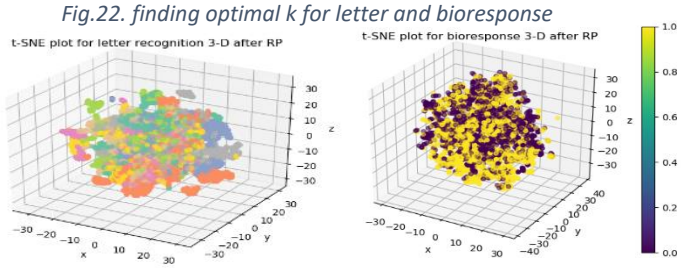


Fig.23.t-SNE plots for letter [LEFT] and bioresponse[RIGHT]

The t-SNE plots for both the datasets after clustering is shown in Fig.23. In Fig.23.b, though number of features is very high, front yellow and behind violet clusters are visible. However, the letter recognition datasets with 26 classes in Fig.23.a does not show enough class-separation in 3-D. The class-cluster alignments of letter recognition and Bioresponse datasets are shown in Fig.24 for k-means and EM. Both algorithms did not cluster well and has very low homogeneity, as almost all clusters have instances of different classes. For example., cluster 1 (red cluster) in Fig.24.a has instances of G, O, Q,A etc, and in Fig.24.b, cluster 1(red) of GMM has A, X and I . The clustering is worse than PCA and ICA presumably because random projections were chosen unlike the systematic PCA. Fig24.c&d and Fig.24.g&h shows the results of both clustering in 2-D feature space for letter recognition and Bioresponse respectively. In a nutshell, random projection performed worse than ICA and PCA for both datasets.

Random forest (RF): Random Forest is implemented using sklearn.ensemble.RandomForestClassifier(). Random Forest is an ensemble learning algorithm which trains several decision trees on various subsets of the data and uses their average. It avoids overfitting as several individual trees cancel each other's bias. The information gain attribute is used by decision trees to get the features with more information and thus

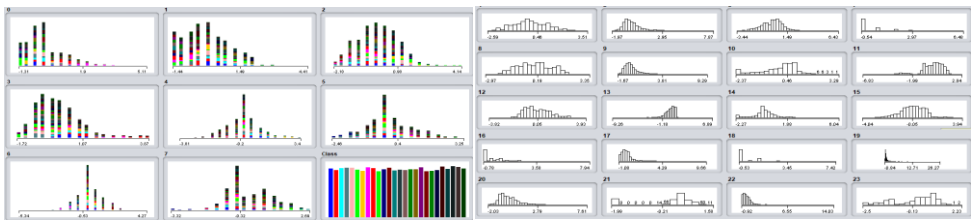


Fig.25.Data in RF spaces for letter [LEFT] and bioresponse[RIGHT]

reducing overall entropy. I used 100 individual trees for this assignment and scree plot to decide optimal number of components as in Fig.26. Based on the values of variance explained, number of components enough to meet the threshold set were selected. For letter recognition dataset, 8 components were chosen as they together account for 93% of variance explained (0.07 is threshold in graph). For Bioresponse, 128 components were chosen as they account for 99.8% of variance explained (threshold is 0.002). The data in new spaces (features by RF) are shown in Fig.25 for letter recognition and Bioresponse datasets. It is noticeable that random forest has discretized the values for each feature. The RF-applied dataset is then clustered using k-means and GMM. In Fig.26, the plots give 15 clusters for k-means and 35 for GMM. For Bioresponse dataset, 5 is optimal for k-means and 6 for GMM. Fig.27 and 28 shows the class-cluster alignment for letter recognition and Bioresponse respectively. In Fig.27, GMM clusters are better as most of the clusters own instances of one class(homogenous), as opposed to k-means where

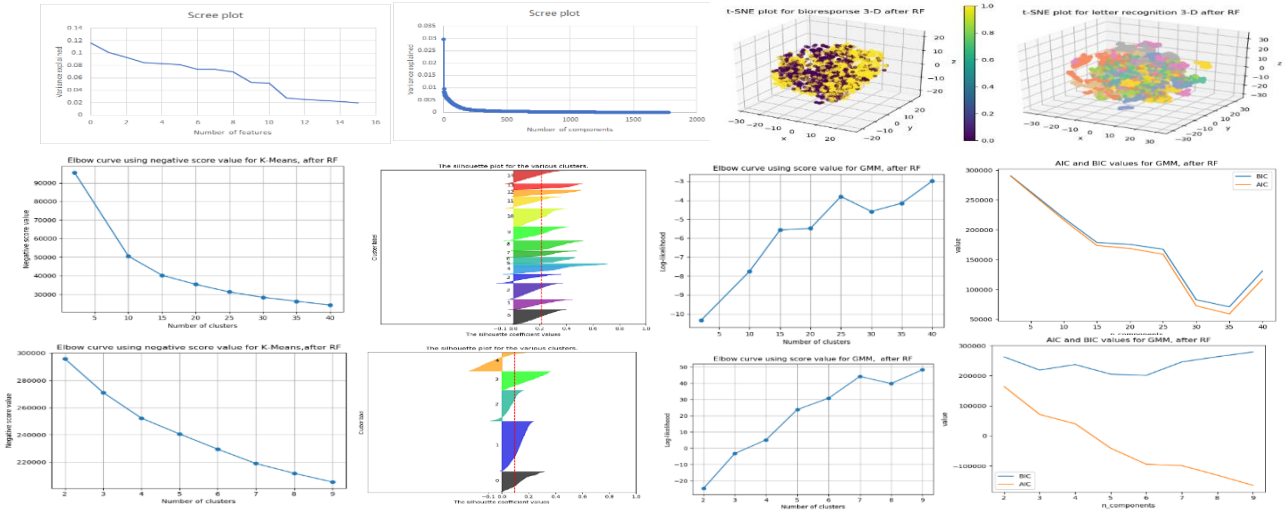


Fig.26.: Scree plots [TOP LEFT] and t-SNE plots[TOP RIGHT]. Finding optimal k for letter [MIDDLE] and bioresponse [BOTTOM]

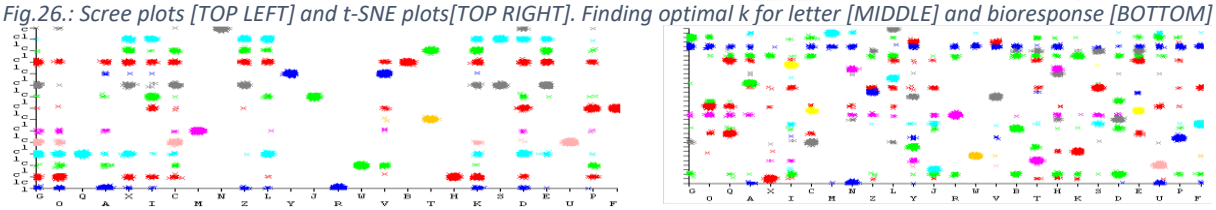


Fig.27.Class-cluster alignment for letter

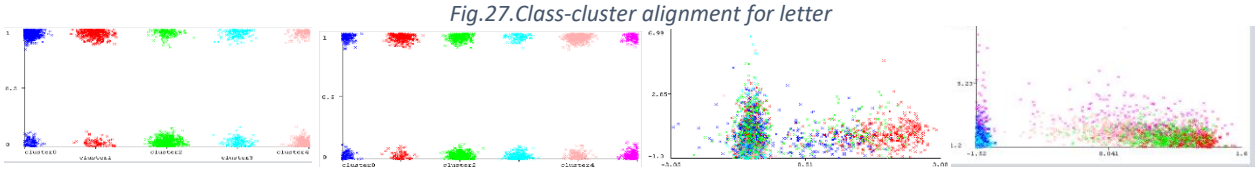


Fig.28.Class-cluster alignment [LEFT] and clustering formed [RIGHT] for bioresponse using k-means [Fig.28.c] and GMM[Fig.28.d]

most of the clusters have instances from multiple classes. In Fig.28, the clusters have instances of both class 0 and class 1. The clustering is better than RP as RF looks for predictive features and not just random features. The t-SNE plots in 3-D for both datasets are shown in Fig.26.c and d. The clusters are not well-separated unlike in PCA &ICA.

Validation and Results of Dimensionality Reduction algorithms: The performance of K-means and EM are validated using metrics such as homogeneity-score(homo) which checks for all clusters, if all the members of a cluster belong to the same class and completeness-score(compl) which evaluates for all classes, if all members of a class belong to the same cluster. V-measure(v-meas), which is the harmonic mean of homogeneity and completeness score is also used. Adjusted Rand Index(ARI) is used to ascertain the similarity between the predicted clustering and the true clustering (actual classes in dataset) by counting sample pairs assigned in the same and different clusters. Adjusted mutual information (AMI) records the information shared between actual labels and the clusters predicted. Higher the value, better the algorithm. For **letter recognition dataset**, overall, GMM with random initialization gives best scores and takes the least time. All DR algorithms gave similar performance unlike in bioresponse, where RF gave good scores than others. For 'random' GMM, the lower optimal k and using random initialization decreased the time cost. Also, it might have started from a favorable random position (randomly chosen) and hence

converged early. ARI is also more for GMM. So, GMM suits the letter recognition dataset than k-Means. K-means use L2 norm to minimize SSE and this makes it biased towards spherical clusters. The intuitive reasoning behind probabilistic soft-clustering working better here can be that it does not depend on L2 norm and the data might be complex geometrical shaped (non-linear), mostly gaussian and not representable by spherical clusters, thus k-means performance getting affected by its bias. Also, GMM prefers gaussian and the features were mostly gaussian. However, silhouette score is more for k-Means because k-means

Without dimenionality reduction

	Algorithm	init	time	homo	compl	v-meas	ARI	AMI	silhouette
	K-Means	k-means++	22.38s	0.450	0.405	0.426	0.166	0.401	0.159
	K-Means	random	31.12s	0.449	0.406	0.427	0.167	0.402	0.114
	GMM	kmeans	20.89s	0.469	0.476	0.472	0.194	0.466	0.023
	GMM	random	14.36s	0.472	0.480	0.476	0.184	0.469	-0.041
PCA									
	K-Means	k-means++	6.20s	0.406	0.455	0.429	0.194	0.404	0.141
	K-Means	random	5.64s	0.384	0.425	0.403	0.181	0.381	0.144
	GMM	kmeans	21.34s	0.542	0.522	0.532	0.219	0.519	0.026
	GMM	random	20.70s	0.529	0.495	0.511	0.233	0.492	0.010
ICA									
	K-Means	k-means++	6.21s	0.391	0.432	0.411	0.191	0.389	0.140
	K-Means	random	7.98s	0.389	0.432	0.410	0.189	0.387	0.130
	GMM	k-means	24.48s	0.536	0.512	0.523	0.236	0.509	0.072
	GMM	random	34.41s	0.552	0.519	0.535	0.245	0.516	0.015
RP									
	K-Means	k-means++	5.32s	0.253	0.316	0.281	0.102	0.251	0.149
	K-Means	random	6.07s	0.257	0.318	0.284	0.106	0.255	0.154
	GMM	kmeans	7.69s	0.410	0.463	0.435	0.181	0.407	0.045
	GMM	random	8.24s	0.422	0.481	0.450	0.179	0.420	0.029
RF									
	K-Means	k-means++	2.06s	0.389	0.485	0.431	0.201	0.386	0.200
	K-Means	random	2.63s	0.389	0.484	0.431	0.200	0.386	0.230
	GMM	kmeans	12.67s	0.391	0.452	0.419	0.102	0.385	-0.014
	GMM	random	7.96s	0.427	0.433	0.430	0.160	0.421	-0.008

Without dimenionality reduction

	Algorithm	init	time	homo	compl	v-meas	ARI	AMI	silhouette
	K-Means	k-means++	6.04s	0.004	0.003	0.004	-0.003	0.003	0.213
	K-Means	random	5.14s	0.004	0.004	0.004	-0.003	0.003	0.238
	GMM	kmeans	6.91s	0.002	0.007	0.003	-0.003	0.002	0.557
	GMM	random	21.59s	0.000	0.000	0.000	-0.000	-0.000	-0.002
PCA									
	K-Means	k-means++	0.20s	0.011	0.009	0.010	-0.000	0.008	0.511
	K-Means	random	0.30s	0.012	0.009	0.010	0.000	0.008	0.516
	GMM	k-means	0.06s	0.000	0.000	0.000	-0.001	0.000	0.381
	GMM	random	0.05s	0.003	0.005	0.004	0.000	0.003	0.502
ICA									
	K-Means	k-means++	1.08s	0.008	0.043	0.013	0.006	0.007	0.020
	K-Means	random	1.23s	0.024	0.014	0.018	0.006	0.014	-0.091
	GMM	kmeans	6.51s	0.011	0.010	0.010	0.006	0.009	0.126
	GMM	random	4.11s	0.008	0.003	0.004	0.003	0.002	-0.071
RP									
	K-Means	k-means++	6.00s	0.014	0.007	0.010	0.005	0.007	0.032
	K-Means	random	5.45s	0.013	0.007	0.009	0.005	0.006	0.030
	GMM	kmeans	7.95s	0.006	0.005	0.005	-0.003	0.004	0.194
	GMM	random	18.48s	0.001	0.000	0.000	-0.000	-0.000	-0.060
RF									
	K-Means	k-means++	0.55s	0.073	0.033	0.045	0.027	0.032	0.094
	K-Means	random	0.72s	0.065	0.029	0.040	0.023	0.029	0.093
	GMM	kmeans	1.13s	0.090	0.037	0.053	0.027	0.037	0.097
	GMM	random	2.44s	0.036	0.016	0.023	0.038	0.016	0.027

Fig.29.Results for Letter[TOP] and bioresponse[BOTTOM]

focus on decreasing SSE and thus implicitly tries to maximize cluster cohesion. Also, the number of clusters formed were mostly high for GMM(average 30) than k-means (average 20) and GMM was more near to actual class count thus helping in improving its shared MI and ARI between true labeling and clusters. For **bioresponse**, overall, the scores are not good except for Silhouette score. This was expected because the dataset is highly susceptible to curse of dimensionality with its 1777 features and 3751 instances. K-means performed comparatively better than GMM. Random GMM might have got stuck in local optima due to random initialization, thus resulting in poor scores. Better completeness and homogeneity score for k-means say that in k-means, comparatively more members in a cluster belonged to the same class than in GMM, however the difference is negligible, and the scores are low. Also, the features are mostly binary which explains why PCA and GMM do not do a good job. The global constraint of mutual orthogonality in PCA and GMM's bias towards gaussian made the scores poor. Though k-means did better comparatively, its score is not good enough, and k-means's bias towards spherical clusters might be the reason because the clusters for this dataset might be geometrically complex shaped and not spherical.

Part 3: Neural network Analysis

The performance of different neural networks trained using preprocessed data is shown in Fig.31 and 32. Using cluster data as the only feature as well as adding it to existing dataset were both experimented. Fig.31 shows the classification accuracy and time complexity when input data was preprocessed by k-means (KM) and EM(GMM). Suffix 1(set1) as in KM1 indicates that cluster was the only feature of NN input data whereas suffix 2(set2) indicates that the existing dataset was augmented with cluster as an additional feature. Set-2 performed far better than set 1. This is because when cluster is the only feature, any mistakes happened in cluster assignments gets passed to NN and information in other features not retained by cluster assignment is discarded. When cluster is just an extra feature, this information contained in other features nullify this to a good extent. When number of clusters is increased, all accuracies improved generally. This is because the underlying dataset has many classes (26) helping to preserve the differences between different classes. GMM gave slightly better performance than KM in set-2 but KM gave better performance than GMM in set-1. So, when cluster is the only feature, the cluster assignments of KM gives more relevant information to NN and when cluster is just an additional feature, GMM's clusters gave slightly better information. All algorithms of Set-2 and

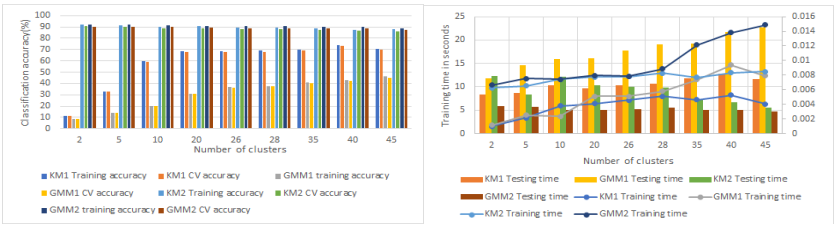


Fig.31.Classification accuracy [LEFT] and time complexity [RIGHT] for k-means and

none in Set-1 gave better performance than Assignment -1 neural network, as shown in Fig.33. Fig.32 shows the classification accuracy when NN input data was processed using different dimensionality reduction algorithms. The training time is very high than testing time for neural networks. In Fig, GMM2 takes the most training time, followed by KM2. Set-2 takes more training time because the features are more, thus increasing the NN complexity. Testing time was comparatively more for Set-1. However, the time complexity was not much different from the assignment-1 neural network, and I guess this is because the number of features were only 16 initially and dimensionality reduction reduces a maximum of 8 features. If the initial features were in thousands and reduced to less a 100, the effect in time complexity would have been more significant. The classification accuracy when input data was preprocessed by DR and then subjected to clustering so that the clusters can be used as the only feature (Fig.32.b) and to augment the existing features (Fig.32.c) are also shown. When the number of features is less, RF gave best accuracy and when the feature count was increased, PCA gave the best accuracy, closely followed by RF and then RP. In both cases , ICA and PCA gave the best accuracy for all clustering algorithm instances, and then RF. RP gave least accuracy throughout the experiment. GMM performed worse than KM for same number of clusters used. This might be because cluster assignments by GMM are less informative for this dataset and as the cluster was the only data fed to NN, NN lost all other relevant information in other features. Also, GMM had showcased a trend of needing more than 26(actual class) clusters for this dataset, and as cluster count increased, GMM performance improved. So, GMM needs more clusters than K-means to represent this dataset, so as not to lose a lot of relevant information. The table below shows that the performance was better than in Assignment1 only when cluster data was used to augment the existing data and no dimensionality reduction algorithm was used and k-means performed slightly better than GMM due to earlier mentioned reasons. This is because information contained in feature asset is lost when dimensions are reduced or when only cluster is fed to NN. But during feature augmentation, NN gets extra information from cluster as well as from the existing dataset.

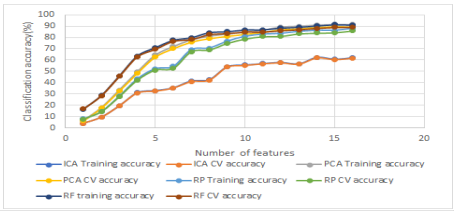


Fig.32.Classification accuracy when input data is preprocessed by DR alone [TOP] and by clustering after DR [BOTTOM]

	Output directly fed to NN				Output used to augment existing features			
	Training accuracy	CV accuracy	Training time	Testing time	Training accuracy	CV accuracy	Training time	Testing time
K-Means Clustering	87.1528	85.5714	8.2356	0.0005	92.1358	90.4800	9.0517	0.0037
EM	37.7198	37.4571	13.8443	0.0042	91.7524	90.2750	14.8854	0.0079
PCA	84.9732	83.0500	5.5933	0.0033				
PCA & K-Means clustering	70.6637	70.1800	6.3570	0.0070	83.8245	82.5700	8.1374	0.0037
PCA & EM	44.3228	44.1600	17.1362	0.0381	84.9079	83.7100	19.9369	0.0059
ICA	56.8754	56.7857	8.8565	0.0092				
ICA & K-Means clustering	71.3714	70.6429	7.8261	0.0074	85.0746	82.9286	9.0127	0.0036
ICA & EM clustering	81.4183	79.8429	6.7175	0.0031	86.5627	84.5071	16.8003	0.0113
RP	80.8757	78.6286	7.5079	0.0042				
RP & K-Means clustering	78.2536	76.5429	8.5966	0.0044	78.2536	76.5429	9.5966	0.0044
RP & EM clustering	79.1063	77.4786	10.9015	0.0089	79.1063	77.4786	11.9015	0.0089
RF	84.3772	82.6500	7.2782	0.0140				
RF & K-Means clustering	81.2306	79.9571	7.1908	0.0037	82.1304	80.5929	9.4462	0.0038
RF & EM clustering	78.3522	76.8929	7.3014	0.0040	81.4621	79.8714	14.2544	0.0099
Assignment 1 NN	87.8370	85.8839	8.0070	0.0056				

Fig.33.Results for neural network

CONCLUSION: In EM, the points midway between clusters are not forced to join one cluster and can belong to several clusters probabilistically, unlike in k-means. EM and k-means can both get stuck in local minima and random restarts can be helpful. k-means is fast and guaranteed to converge unlike EM with infinite configurations in the probability space. But for practical purposes, EM do converge mostly. K-means is easier to interpret compared to EM. K-means suits spherical data due to its bias caused by dependence on L2 norm, while EM is good for complex geometrical shaped data(non-linear), mainly gaussian data. EM uses all accessible components, so cluster initialization is difficult with high dimensional data, so k-means is better for high dimensional data, which explains why it was better than GMM for bioresponse. When data has non-Gaussian components, k-means may be better than EM as in letter dataset. For performance improvement, the 'init' parameter in k-means can be set as 'k-means++' to speed-up convergence in a smart way, instead of random. PCA looks for global mutual orthogonality with maximum variance. RP is fast and easy to interpret. ICA looks for independent features and RF looks for predictive features.

REFERENCES:

1. P. W. Frey and D. J. Slate. "Letter Recognition "., 1991. Retrieved from https://www.openml.org/d/6
2. Ingelheim, B. "Bioresponse".,2015. OpenML. Retrieved from https://www.openml.org/d/4134
3. Tay,J.,CS-7641-assignment-3.GitHub. Retrieved from https://github.com/JonathanTay/CS-7641-assignment-3