UNSUPERVISED LEARNING AND DIMENSIONALITY REDUCTION REPORT

DATASETS UTILIZED

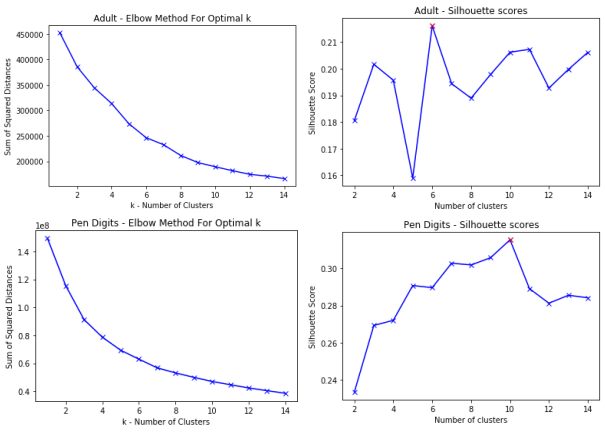
In this assignment we will explore unsupervised learning & dimensionality reduction algorithms. In doing so, we utilize the same datasets obtained for Assignment 1.

1. The first is the *Census Income Data Set* taken from the UCI Repository of a 1994 census data to predict whether the income of an adult exceeds $50k/year which is interesting because it is based on several attributes such as age, work class, education, occupation, sex, native country, and others. This dataset comes with 48842 different instances which can be used for the binary classification task.
2. The second is the *Pen-Based Recognition of Handwritten Digits Data Set* taken from the UCI Repository of handwritten digit samples from 44 writers. This is a multiclass dataset with 16 attributes and 10992 instances which can be used for the multiclass classification task. It is interesting for its practical implications as well as its application for optical character recognition which is a computer vision problem.

1. CLUSTERING

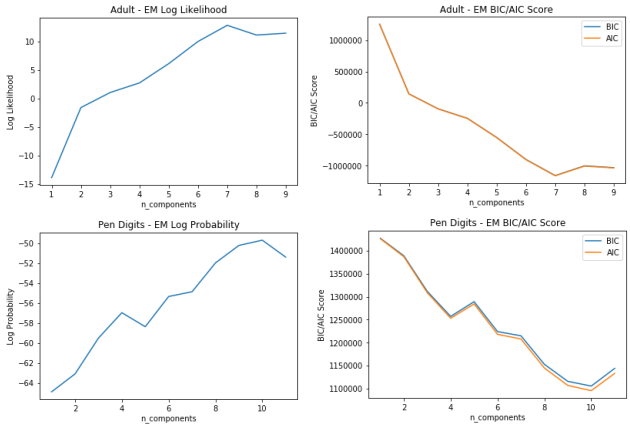
In this first experiment we will perform clustering on both our datasets using the *k*-means clustering and Expectation Maximization algorithm. The first algorithm implemented is the ***k*-means** clustering algorithm. This algorithm performs clustering in our data by initially choosing *k* centers at random, assigning each point to the closest center, recomputing the centers by averaging the clustered points, and repeating the last two steps until convergence. The second algorithm is **Expectation Maximization**, that unlike *k*-means, creates "soft" boundaries between clusters, where a point with a certain probability is assigned to a gaussian that generated it. This is performed by calculating the *expectation* or probability of each point being generated by a component of the model and then tweaking the parameters to *maximize* the likelihood of the data given the assignment. While performing clustering, the first thing we have to determine is the value of *k* for our data, i.e. how many clusters are present. Ideally, we would expect that the number of classes match the number of clusters for our datasets, however this may not be guaranteed as the points may be clustered differently. Also, we have to determine our measure of distance/similarity used while assigning points to the closest centers. Let's experiment by running these algorithms on both datasets and analyze the results we obtain. We would like to note that we tried two distance metrics, the Manhattan and Euclidean distance, however both gave similar results, so we stay with the Euclidean distance metric for the rest of this report which is the default for `sklearn` which we will use for the remaining parts of this assignment.

K-MEANS



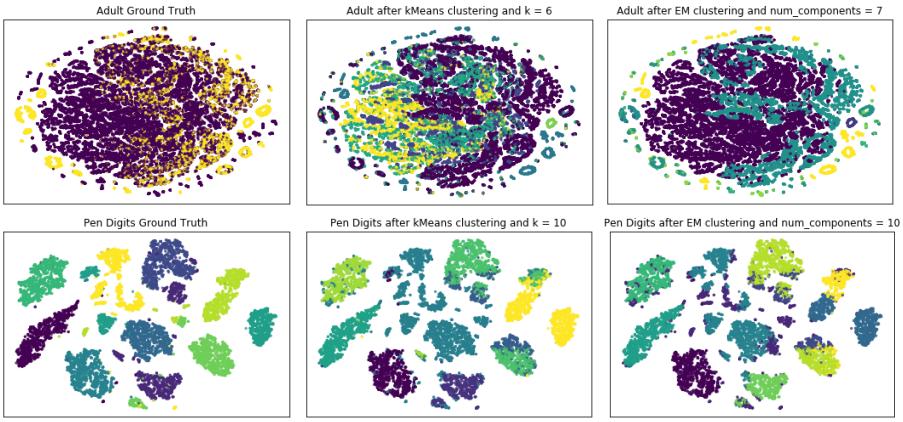
We first perform k-means clustering on this dataset, which tries to separate samples in groups with equal variance minimizing the inertia or within cluster sum of squares. We want to first determine what is the optimal *k* (number of clusters). One method to determine this is using the elbow method. We will plot the within cluster sum of squared error which describes the explained variances against the number of clusters. Smaller cluster numbers have high error (not explaining a lot of variance) but as the cluster number increases, the error decreases, giving an angle in the graph after which there is only marginal gain. We choose the number of clusters at this (elbow) point. In some cases, the elbow is not very evident as in the plots to the right for both datasets, so we use the silhouette analysis to study the separation distance between clusters with values ranging between `[-1, +1]`, where +1 indicates the sample is far from neighboring clusters, `0` indicates that it is on or close to the boundary, and negative values indicates samples that might be assigned to wrong clusters. We want the highest silhouette score to get the optimal k. As we can see in our plot fork the Adult dataset, the optimal value for k-means clustering is `k = 6`. For the Pen Digits dataset, the number of clusters is `k = 10`.

Expectation Maximization



We now perform EM on both our datasets using the `GaussianMixture` implementation in `sklearn`. We first want to determine the optimal number of gaussian components (corresponding to clusters). The Bayes Information Criterion (BIC) and the Akaike Information Criterion (AIC) can be used to select the number of components in a Gaussian Mixture. With these metrics, we can recover the number of components given much data is available and assuming it was actually generated from a mixture of Gaussians. The optimal number of clusters is obtained when the BIC/AIC is minimum. Similarly, we can also plot the Log Likelihood vs number of components and look for an elbow or maximum point, which would correspond to the optimal number of clusters. As we can observe from the image, the optimal number of clusters for the Adult dataset is $k = 7$ when Log Likelihood is maximal, and BIC/AIC is minimal. Similarly, for the Pen Digits dataset, the optimal number of clusters is $k = 10$.

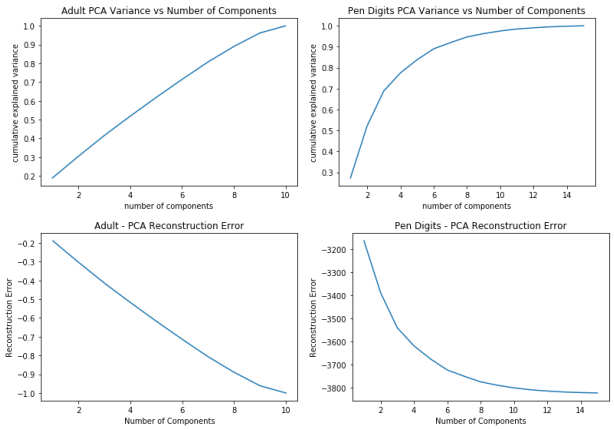
Clustering Results



Now let's visualize the results of clustering to see how well these algorithms did. We use t-SNE, a tool to visualize high-dimensional data in `sklearn` that reduces our dataset in 2D. In the images we have first plotted the ground truth labels for both the Adult and Pen Digits dataset. Beside we show the results of k-means and EM clustering. For the Adult dataset we can observe that both algorithms get clusters that line up with the ground truth labels. K-means does a decent job separating the clusters in a way that match the labels. As we can see the clusters do not correspond exactly to the true labels, but each cluster separated the data based on certain characteristics, i.e. the big purple cluster was clustered as smaller yellow and green clusters. EM also does a decent job, despite having more clusters than true labels, each of these components manage to separate the purple and yellow clusters present in the ground truth, as a big purple cluster and multiple smaller clusters (with different colors) for the rest. The clustering seems to line up because each person has certain characteristics and those common characteristics form multiple clusters, that when put together, match the true labels. For the Pen Digits dataset, as we can observe, the ground truth consists of 10 islands with no overlap, each being a digit. From the results of k-means and EM clustering, we can see that both tried to classify the dataset into 10 clusters, and they overall do a decent job. Clustering does not guarantee that we will get the "labels" we are looking for, but some clustering is returned. In this case maybe some characteristics of specific digits are grouped together, and since most digits have distinct characteristics, most islands have been classified as distinct clusters. However, we can also see some clusters that are separate in the ground truth, but are joined together after clustering, which backs our claim that clustering does not necessarily return the labels, but clusters with similar characteristics.

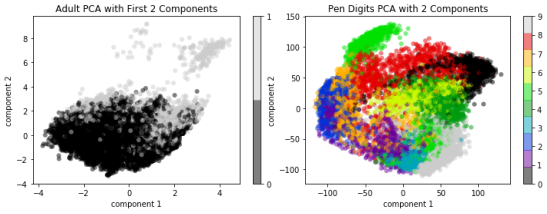
2 & 3. Dimensionality Reduction & Clustering

Principal Component Analysis (PCA)

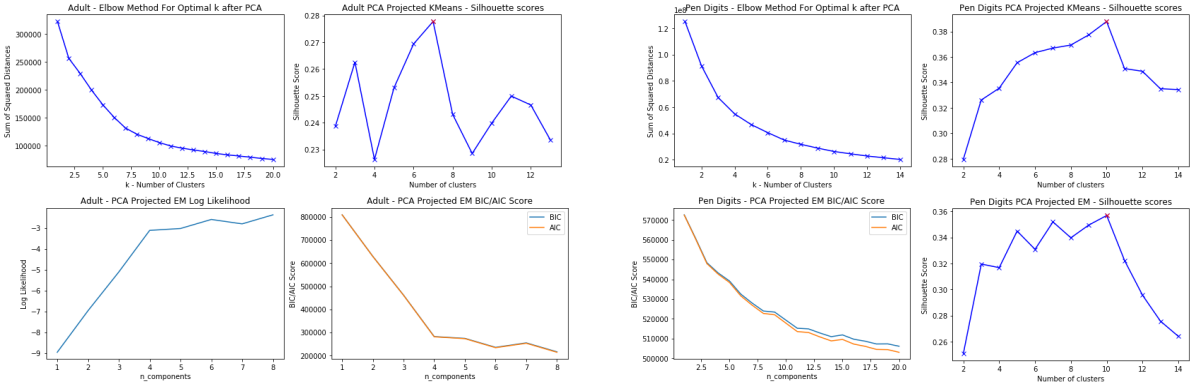


The first dimensionality reduction algorithm that we implement is PCA, which decomposes the features as a set of orthogonal components explaining the variance. It's about eigenvectors and eigenvalues recovering structure in our data (assuming it's numeric) finding the direction (principal component) of maximal variance. It has one important property, if we were to get rid of some of the dimensions, and we make certain they are with the lowest eigenvalues (variance), we are guaranteed

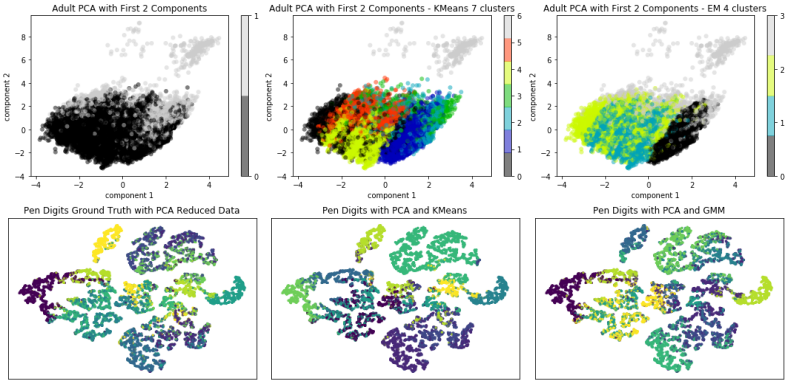
that this subspace maximizes reconstruction. This is what we will use to find the optimal number of components. We make a variance vs number of components plot, where we can observe that the distribution of eigenvalues increases as the number of components increases. Ideally to find the optimal number of components, we want one that can capture more than 70% of the variance. For the Adult dataset by using the first 6 principal components we explain ~70% of the variance, and for the Pen Digits dataset by using the first 5 components, we can explain ~80% of the variance, so we choose these as the optimal number of components for each dataset. So now we zero out the remaining smallest principal components, resulting in a lower-dimensional projection of the data preserving the maximal variance. We can also observe that the reconstuction error decreases as the number of principal components increases.



Now we will describe how the data looks in the new space we created using PCA. We have found the optimal stretch and rotation of the higher dimensional space that allows us to see the data along the directions with high variance in an unsupervised manner (without using the labels). For the Adult dataset we can see that there is some overlap between the two classes, but the black cluster is concentrated near the bottom and the gray cluser towards the top. For the Pen Digits dataset, we can see that there is some overlap between certain digits, while some others are more distinct. PCA has reduced the dimensionality, making the clusters somewhat more evident which seems could help in the clustering experiment. Let's reproduce the clustering experiment and describe our results. Below we can see the plots for determining the optimal number of clusters using k-means and EM on the PCA reduced data.

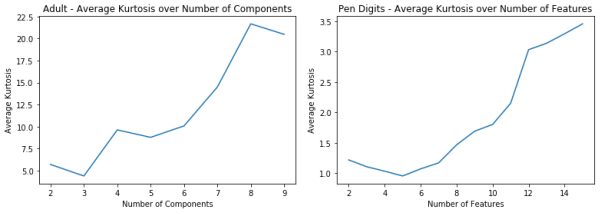


On the left we can observe the plots for the Adult dataset. For k-means, using the elbow method in the SSE plot, we can see a slight bend at `k = 7`. We run the silhouette analysis, and reconfirm that the optimal number of clusters is 7. For EM, we in the log likelihood plot, we see a bend at 4 components. Similarly, the the BIC/AIC plot, we se a bend at 4, which suggests is that `k = 4` is the optimal number of components for EM. On the right we observe the plots for the Pen Digits dataset. For k-means, using the elbow method in the SSE plot, the bend is not very evident. We run the silhouette analysis, and observe that the optimal number of clusters is `k = 10`. For EM, in the log likelihood plot, we see that as the number of components increases the BIC score decreases (and likelihood increases). Since no elbow is evident, we run the silhouette analysis and see a peak at `k = 10` which is the optimal number of components for EM. The number of clusters obtained after PCA for Pen Digits are similar to what we got without dimensionality reduction, but different for the Adult dataset. Let's visualize the results of clustering. We can see that for the Adult dataset, k-means and EM both do a decent job. Though having more number of components than true labels, the big black cloud at the bottom on the ground truth plot has separated into multiple smaller clusters, and the gray cloud at the top also has been clustered separately by k-means and EM. For the Pen digits dataset, the ground truth labels show some islands, where some digits have similar characteristics to others, so a boundary is not very clearly defined. Beside we can see that k-means tries to identify clear islands as clusters, but because there is some overlap, the clusters are not very well defined. On the other hand, because EM assigns each point with certain probability to each cluster, allowing for "soft" boundaries, it does an overall great job clustering compared to the ground truth plot.

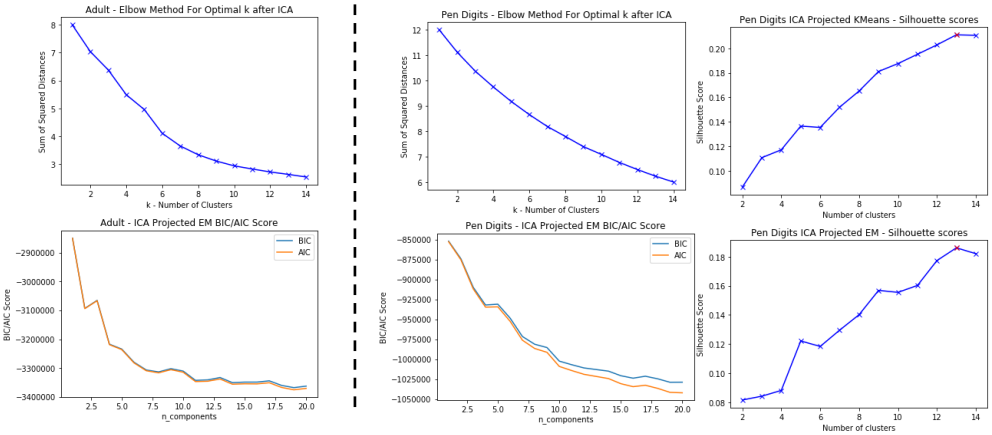


Independent Component Analysis (ICA)

The next dimensionality reduction algorithm that we implement is ICA. We operate in a world where things cause other things and we want to recover the causes. There are true hidden things out there in the world that give raise to variables we can observe (cause things). They are *independent* and we assume that they are *discriminative* (distinguishes a feature from other features). PCA doesn't do any of this because it maximizes variance via the gaussian (normal) distribution. ICA is an alternative to PCA that finds projections statistically independent from each other. It seeks to capture the data we originally have, while maximizing the independence between them. It turns out that it maximizes kurtosis and assumes the data is non-gaussian. ICA doesn't work well when the data is gaussian, in that case PCA is used. However, not everything has an ICA, and another difference is that eigenvalues in PCA have order, while the independent components of ICA have no order. We use the `FastICA` implementation from `sklearn` and we plot the Average Kurtosis over the Number of Components. We want to maximize the kurtosis to get the most information after performing ICA. From the plot to the right we can observe that for the Adult dataset using 8 independent components we get maximum kurtosis, and for the Pen Digits dataset, using all 14 independent components we get maximum kurtosis. However, we can see a knee at 12. Since the purpose of this experiment is to reduce dimensionality and while experimenting with 12 and 14 features, the results were mostly the same, we choose 12 features for this report. We can observe that as we increase the number of features, the distributions become more kurtotic.

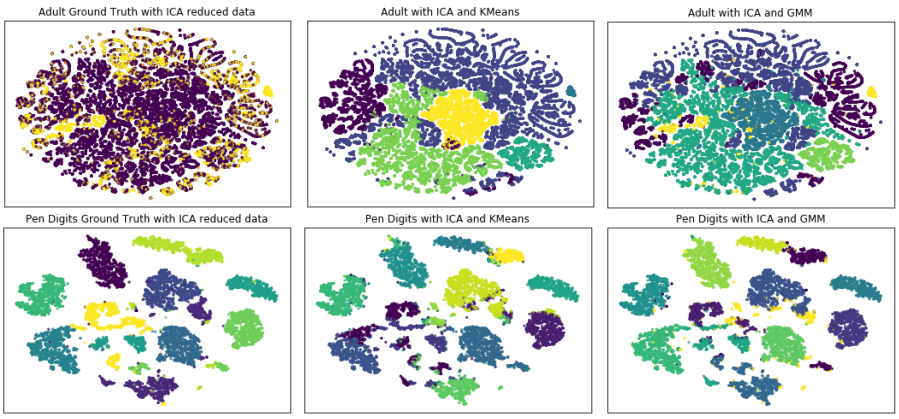


Now we will describe how the data looks in the new space we created using ICA and also reproduce the clustering experiment and describe our results. Below we can see the plots for determining the optimal number of clusters using k-means and EM on the ICA reduced data.



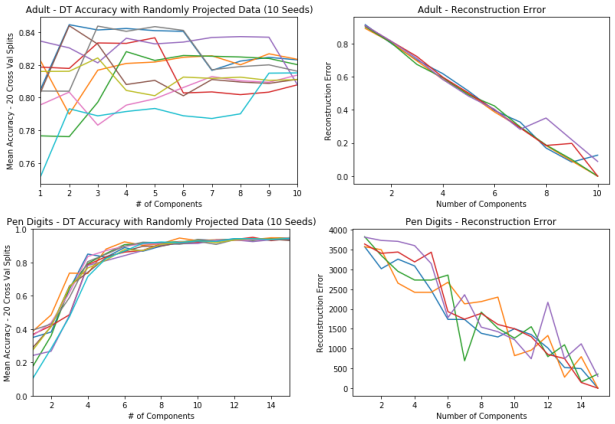
On the left side of the image, we can observe the plots for the Adult dataset. For k-means, using the elbow method in the SSE plot, we can see a bend at $k = 6$ which will be the optimal number of clusters. For EM, we in the BIC/AIC plot, we see a bend at 6 as well, which suggests is that $k = 6$ is the optimal number of components for EM. On the right we observe the plots for the Pen Digits dataset. For k-means, using the elbow method in the SSE plot, the bend is not very evident. We run the silhouette analysis, and observe that the optimal number of clusters is $k = 13$. For EM, in the BIC/AIC plot, we see that as the number of components increases the BIC score decreases (and likelihood increases). Since no elbow is evident, we run the silhouette analysis and see a peak at $k = 13$ which is the optimal number of components for EM. The number of clusters obtained after ICA for Pen Digits are higher to what we got without dimensionality reduction and PCA meaning that even more smaller characteristics from digits are being grouped together. For the Adult dataset it is mostly close to the number of clusters obtained with PCA and without dimensionality reduction.

Let's visualize the results of clustering. We can see that for the Adult dataset, k-means clustering does not line up with the true labels. We can see from the ground truth plot that there is overlap, and k-means tries to group everything into distinct groups; the clustering achived is different from the true labels. However EM does an acceptable job, as the clustering lines up with the true labelling

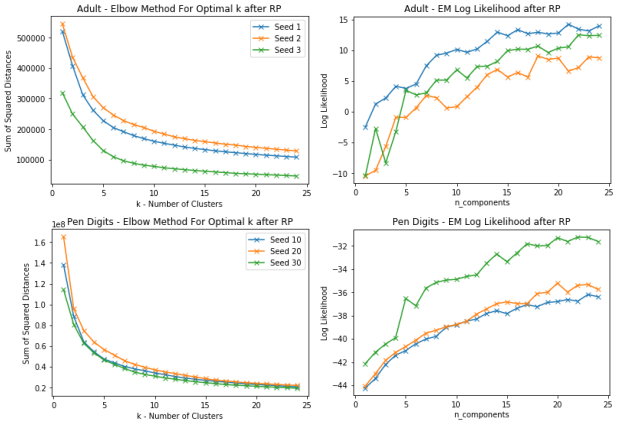


with the difference that the bigger cluster is sparated into several smaller clusters. For the Pen Digits dataset, the ground truth labels show some islands, where some digits have similar characteristics to others, with a clearly defined boundary. Beside we can see that k-means and EM both try to identify clear islands as clusters and do an overall decent job compared to the true labels. The results of clustering arent so good as using PCA, so this hints that the projection axes for ICA do not seem to capture something very meaningful for both datasets.

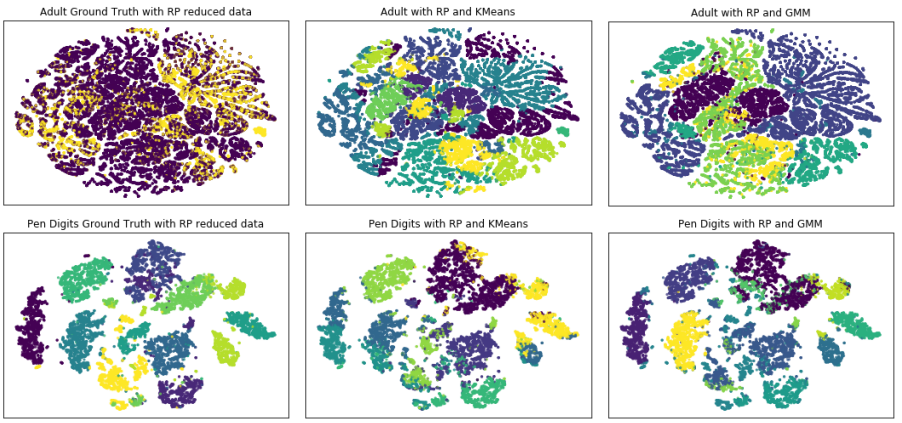
Random Projections



The third dimensionality reduction algorithm we implement is Random Projection which performs random components analysis by randomly generating directions and projecting the data into those dimensions while maintaining some of the correlations from the remaining directions. The main advantage of RCA is that it is fast. We use the `sklearn`'s Sparse Random Projection implementation that reduces the dimensionality by projecting the original input space on a sparse random matrix. Now an input we must tune is the number of features to use for performing Random Projections. One method to do this is to implement a base learner and keep reducing the number of features until we see a drop in accuracy. We implement a simple decision tree using `sklearn` and perform multiple runs with different seeds to get the optimal number of components. For the Adult dataset we would like to note that the plot is a zoomed in version of the accuracy plot. This plot is such because some features are randomly selected, sometimes the noise is let in and sometimes the true signal is left in. Therefore, some runs are better and some are worse, and they converge as the features approach the full set. We choose 8 as the number of components as around that point the variance is minimal between the multiple runs. For the Pen Digits dataset, we choose 10 as the number of components. Alongside we also plot the reconstruction error for each dataset. From the plots we can observe a linear behavior: the more components we include, the lower is the reconstruction error.



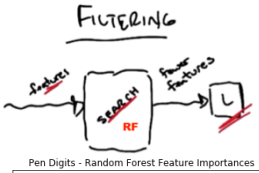
Now we will describe how the data looks in the new space we created using RP and also reproduce the clustering experiment and describe our results. On the right we can see the plots for determining the optimal number of clusters using k-means and EM on the RP reduced data. As we can observe, we did 3 runs with different seeds and take the results that are combined between runs. On the left we can observe the plots for the Adult dataset. For k-means, using the elbow method in the SSE plot, we can see an elbow at `k = 10`. For EM, we in the log likelihood plot, we see a bend at `k = 6` as the optimal number of components. We are getting different number of clusters than we did before, which seems to indicate that even different characteristics of the adults are being clustered together. On the right we observe the plots for the Pen Digits dataset. For k-means, using the elbow method in the SSE plot, we can see an elbow at `k = 7`. For EM, we in the log likelihood plot, we see a bend at `k = 12` as the optimal number of components. We are getting more clusters than we did before, which seems to indicate that even more finer characteristics of the digits are being clustered together.



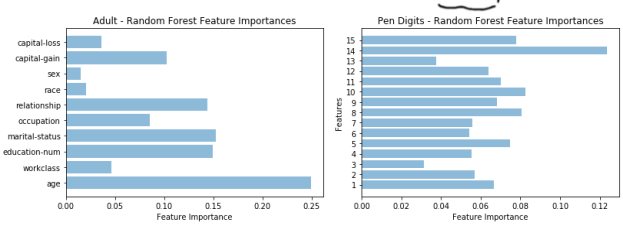
Let's visualize the results of clustering. We can see that for the Adult dataset, k-means and EM both do a decent job. Though having more number of components than true labels, the big purple cluster and yellow cluster on the ground truth plot has separated into multiple separate smaller clusters by k-means and EM. For the Pen digits dataset, the ground truth labels show some islands, where some digits with

similar characteristics clustered together, with clearly defined boundaries. Beside we can see that k-means tries to identify clear islands as clusters, though k-means groups together some clusters that are originally separate, and those that were separate have been clustered together. However the majority has been clustered spearately. Similarly, EM obtains a similar clustering to k-means however unlike k-means does an overall better job clustering compared to the ground truth plot.

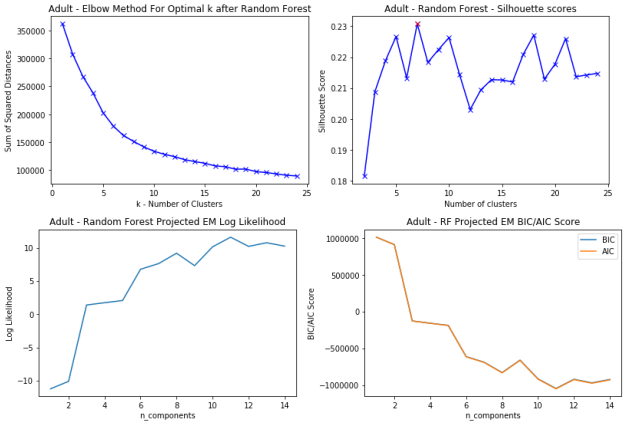
Random Forest Dimensionality Reduction

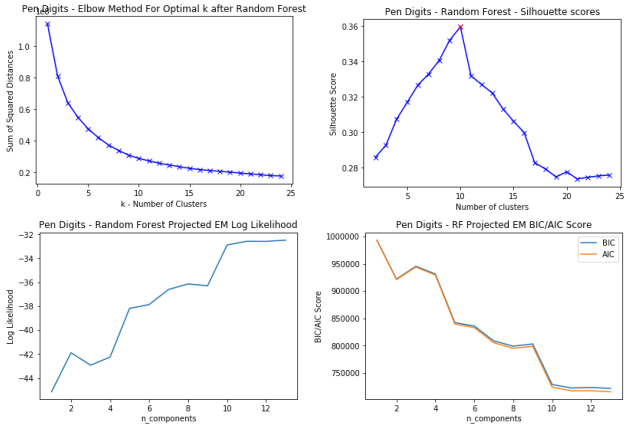


Unlike PCA, ICA, and RP that perform feature transformation, the random forest approach performs feature selection for dimensionality reduction. We perform forward search, add a feature at a time (information gain) and keep building. We build a model of a random forest of decision trees, and we end with a bag of features that we will use. The downside of this approach is that we do not take into account model bias of Neural Networks which we will use in the next experiment. This might work for the forest of decision trees (because that's the bias we include) but might not work for NN or some other algorithm. However, we gain speed by performing this approach. So, we run this implementation and we obtain the importance for each feature. We can see this plotted in the bar chart above to the right. For the Adult dataset we can observe that age is the feature with most importance, followed by education and marital-status. To get the features which are most important, we choose those that are $\geq [mean(importance) - \frac{1}{2} * std(importance)]$. Performing this calculation, we end up with 10 → 8 features for the Adult dataset and 15 → 10 features for the Digits Dataset.

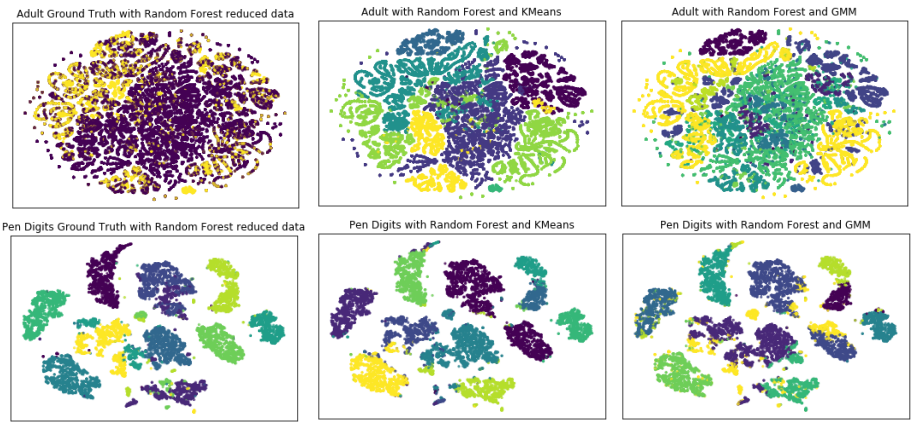


Now we will describe how the data looks in the new space we created using RP and also reproduce the clustering experiment and describe our results. Below we can see the plots for determining the optimal number of clusters using k-means and EM on the RP reduced data.





On the left we can observe the plots for the Adult dataset. For k-means, using the elbow method in the SSE plot, we can see an elbow at `k = 7` which we reassure by performing the silhouette analysis which gets us the same number. For EM, we in the log likelihood plot we see a peak at 11 and in the BIC/AIC plot a min value at `k = 11` which is the optimal number of components. On the right we observe the plots for the Pen Digits dataset. For k-means, using the elbow method in the SSE plot, we can see an elbow at `k = 10` which we reassure by performing the silhouette analysis which gets us the same number. For EM, in the log likelihood plot and BIC/AIC plots, we see `k = 10` as the optimal number of components. The clustering numbers we are getting are similar to when we performed PCA.

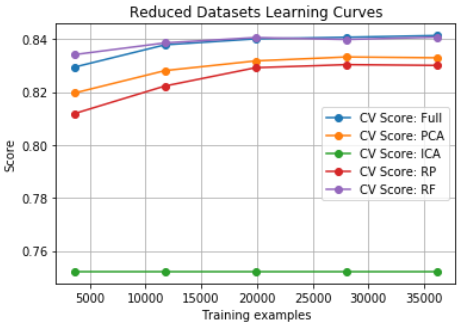


Let's visualize the results of clustering. We can see that for the Adult dataset, k-means and EM both do a decent job. Though having more number of components than true labels, the big purple cluster and yellow cluster on the ground truth plot has separated into multiple separte

smaller clusters by k-means and EM that light up with the true labels. For the Pen digits dataset, the ground truth labels show some islands, where some digits with similar characteristics clustered together, with clearly defined boundaries. Both k-means and EM are able to easily identify each island as a separate cluster for the majority of the points matching the true labels.

4. Neural Networks after Dimensionality Reduction

We now apply the dimensionality reduction algorithms to the Adult dataset from Assignment #1 and rerun our neural network learner on the newly projected data.



Algorithm	Train Accuracy	Test Accuracy	Wall Time
Original Dataset	83.97	84.56	13.20 s
PCA	83.37	83.27	10.40 s
ICA	75.10	75.65	2.97 s
Randomized Projection	83.41	83.43	13.00 s
Random Forest reduced	84.14	83.75	9.40 s

We can observe that all our dimensionality reduction algorithms have a faster wall time for training the neural network compared to the base neural network from Assignment 1. Surprisingly all dimensionality reduced datasets except ICA obtain an accuracy of about 83% closer to what the base NN obtains of about 84%. This leads us to believe that the data contains gaussian noise for which ICA could not successfully extract the independent components, hence led to bad results after dimensionality reduction. On the other hand, PCA, RP and RF achieve high accuracy but with varying training times. From the previous sections, we observed that some algorithms had more features than others, and those that had lesser features achieved faster training times. This small experiment shows us that dimensionality reduction works for combating the curse of dimensionality. We can get comparable results with a reduced feature set, and in a faster time. I'd like to note that the results are not very significant given that we did not have too many features, however this may work very well when we are working with very high dimensional space such as images, where there are many more features.

5. Neural Networks with Clustering as Dimensionality Reduction

For this experiment, we apply the clustering algorithms to the same dataset to which you just applied the dimensionality reduction algorithms treating the clusters as if they were new features (like a dimensionality reduction algorithm). Below in the table we present the result obtained.

Custer as Only Attribute	Train Accuracy	Test Accuracy	Wall Time
Original Dataset	83.97	84.56	13.20 s
PCA k-means clusters	75.22	75.15	5.69 s
PCA EM clusters	75.66	75.96	3.33 s
ICA k-means clusters	75.19	75.29	2.34 s
ICA EM clusters	75.27	74.98	2.10 s
RP k-means clusters	75.12	75.58	2.48 s
RP EM clusters	75.29	75.29	2.75 s
RF k-means clusters	75.21	75.22	10.70 s
RF EM clusters	75.03	75.92	9.21 s

We can observe that adding clusters as the only attributes, the number of features is significantly less, therefore the training times have reduced a lot compared to using all the data. However, the accuracy scores for all clusters after dimensionality reduction were about ~75%. This is much lesser compared to the ~84% obtained by using the entire dataset. This shows us that clusters alone do not help predict the labels for a dataset, and this mainly because the clusters obtained may not necessarily match or represent the true underlying labels. This may also be happening because each true label is getting split into multiple clusters and clustering alone is not giving enough information to the learner. However, it does better than chance, so all we can say is that within the cluster lies some information, but it is not enough to get a more accurate prediction.

We also explore by adding the cluster as an additional attribute to the dimensionality reduction algorithms and see what the outcome was.

Cluster as Addition Attribute	Train Accuracy	Test Accuracy	Wall Time
Original Dataset	83.97	84.56	13.20 s
PCA + k-means	83.55	83.52	9.94 s
PCA + EM	83.00	83.67	9.11 s
ICA + k-means	75.11	75.61	1.99 s
ICA + EM	75.26	74.90	3.31 s
RP + k-means	83.29	83.38	16.00 s
RP + EM	83.07	83.30	14.2 s
RF + k-means	84.32	83.86	9.07 s
RF + EM	84.34	84.02	11.10 s

As expected, since there is one more attribute, the training time is a bit higher than while not including it. However, it is not significantly higher as it is only one feature. On the other hand, we can see that the accuracy scores are a bit higher compared to not using the cluster alone, and however not completely close to the ~84% accuracy obtained by using all features, it does a decent job of getting to ~83% accuracy. This once again backs up that clustering conveys some information however it is not enough to increase our learner's accuracy significantly.

CONCLUSIONS

CLUSTERING

Through this assignment we have got exposure to two unsupervised learning algorithms for clustering. We compared the results of using the Euclidean distance and the Manhattan distance metrics initially and the results were not significantly different. Therefore, we utilized the Euclidean distance metric for this report which is the default metric for `sklearn`. It would be interesting to experiment with different distance metrics in more depth, but due to time constraints we limited the choice to just one. These are our main takeaways for each clustering algorithm:

K-MEANS CLUSTERING

- We try to cluster data into groups with equal variance minimizing the within-cluster sum of squares criterion
- This makes an assumption that clusters are convex and isotropic which does not always occur; hence we end with multiple clusters when the shapes are elongated or irregular, despite them being part of the same cluster. This problem was often encountered in the Pen Digits dataset where sometimes the elongated cluster in ground truth label was split into two separated clusters after k-means
- It sets hard boundaries between clusters and performs poorly when there is overlap between clusters
- Given sufficient time it converges, however we can get unlucky and get to a local minimum
- To combat some of these problems of overlapping boundary and cluster shapes, we should use EM instead.

EXPECTATION MAXIMIZATION

- This algorithm performs clustering by assigning each sample to the Gaussian that it most likely belongs to.
- This allows shapes such as ellipsoids to group clusters that are not convex or isotropic
- It sets soft boundaries between clusters and performs very well when there is overlap between clusters as in the Adult dataset
- Repeating the E and M steps we can say that as the number of iterations increase, it does not diverge so we can assume that it "converges" to a local optimum
- k-means is actually an equivalent to EM algorithm with a small equal diagonal covariance matrix.

In general, for both datasets the clusters obtained mostly lined with true labels, however not with the exact same number of labels as the cluster numbers. We could notice that each label was split into multiple smaller clusters based on specific characteristics found within the data. We can say that clusters do not necessarily represent the labels but rather some grouping that exists within the data. One change we might make to each of those algorithms to improve performance is to experiment with several distance metrics and see how it performs, because in very high-dimensional spaces, Euclidean distances tend to become inflated. In the Adult dataset, the nature of the data was such that there was a bunch of overlap

between the clusters for which in general EM did better than k-means. For the Pen Digits dataset however, there were clear groups corresponding to each digit for which k-means and EM both generally performed very well.

Dimensionality Reduction

From our experimentation we can conclude that dimensionality reduction is indeed very helpful to overcome the curse of dimensionality problem. We can identify important features from the entire feature set and then run our algorithms and get comparable results to training on the entire dataset in less time, as seen in our neural network experiments. Below are the takeaways for each algorithm implemented:

Principal Component Analysis

- Is a feature transformation method that It tries to find the direction of maximal variance via the principal components.
- This works in such a way that we can maximize the reconstruction of the data with minimal loss. We end up with new features where each represents correlations between the previous features.
- It is very fast.
- It always exists in most of the data, and it is particularly used when the data is gaussian.
- PCA does not assume that sometimes our data is independent and discriminative and because PCA maximizes variance it performs poorly in that case. In those cases, we can improve by using ICA.
- For our datasets we could see that as the number of components increased, the distribution of eigenvalues increased (variance increased), and the reconstruction error was minimized.

Independent Component Analysis

- PCA adds all the independent variables together and to be discriminative it needs high kurtosis. To overcome this problem of independence and discrimination, ICA is used as it can detect the “independent” components from our data.
- It seeks to capture the data we originally have, while maximizing the independence between them.
- It assumes that the data is non-gaussian for it to perform well. This is one of the main reasons why we got poor performance in our datasets. The nature of the Adult and Pen Digits dataset is such that the data is gaussian (with some noise) therefore ICA had a hard time extracting the independent components. Therefore, the projection axes for ICA did not capture anything meaningful for our datasets
- It is not as fast
- For our datasets we could see that as the number of features increased, the distributions were more kurtotic.

Random Projections

- Random Projection which performs random components analysis by randomly generating directions and projecting the data into those dimensions.
- It works well in practice because while performing the random projections, it maintains some of the correlations from the remaining dimensions.
- The main advantage of RCA is that it is very fast.
- For our datasets we could see that as the number of features increased, the reconstruction error was minimized.
- In experimentation on our datasets, each run gives a different set of random features (lot of variation), so in some cases we may expect that a random projection from one run may work better than with another run. Therefore, in practice multiple runs should be tested to get a better performance.

Random Forest Dimensionality Reduction

- Unlike PCA, ICA, and RP that perform feature transformation, the random forest approach performs feature selection for dimensionality reduction. We perform forward search, add a feature at a time (information gain) and keep building.
- The downside of this approach is that we do not take into account model bias of the final learner we will use
- However, we gain speed by performing this approach.