Christopher Sullivan

Professor O'Neill

CS-390

4/15/23

Unsupervised Learning: Clustering and Dimensional Reduction

Introduction

Problems

During the course of this study, various unsupervised learning experiments were conducted on multiple data files. To begin, two separate datasets were transformed into sets of clusters using two unique clustering algorithms. Following this clustering, the datasets then underwent the process of dimensional reduction by three reductional algorithms. Furthermore, the newly transformed datasets were again clustered by the two clustering methods, and the results compared. Lastly, the output matrices of all five reduction and clustering algorithms were individually fed to *scikit-learn*'s neural network implementation, which had previously classified one of the datasets. The results of the initial classification were then compared to the results following dimensional reduction.

Relevant Data

In order to demonstrate the strengths and weaknesses of each algorithm, two unique datasets were used in the study. The datasets in question are both binary classification datasets where each record is labelled with a class value of 0 or 1. The first dataset, *emails.csv*, is a set of 5,172 records of e-mails, each with 3,000 recorded features representing the respective occurrences of the most common words across all the e-mails. This highly-sparse dataset is classified by whether or not the e-mails were listed as spam. In addition, the next dataset

diabetes.csv is a table of 768 records representing Pima Indian women. Each record in this table possesses 8 features pertaining to health and physical background, and is classified by whether or not the patient tested positive for diabetes. Overall, the combination of these two datasets is a good case for analyzing the traits of clustering and dimensional-reduction algorithms; while they both possess substantial instance counts, *emails.csv* has a far larger feature and instance count, which will certainly affect the outcome of the algorithms in comparison to *diabetes.csv.*

Initial Clustering

To commence analysis, both *emails.csv* and *diabetes.csv* were passed to *scikit-learn*'s K-Means Clustering algorithm. In order to quantify the effectiveness of the algorithm, the silhouette scoring formula was applied to the models which it produced. The silhouette score is a formula which takes into account the average distance between all points within a cluster, as well as the average distance between clusters altogether, thus measuring the cohesion of the cluster space created by an algorithm. Generally, the closer the silhouette score is to its' upper bound of 1, the more distinctly partitioned the cluster space is. Upon execution of the algorithm on the data, it became apparent that more K-Means clusters led to lower silhouette scores for both the emails and diabetes datasets. Indeed, the minimum value of 2 clusters appeared to produce the most cohesive partitioning, with scores only decreasing as cluster count incremented. While this is significant, it should be taken into consideration that this may be due in part to the calculation of the silhouette score. With a rising number of clusters in a finite space, it is possible that the magnitude of the decline is simply due to the shrinking average distance between clusters.

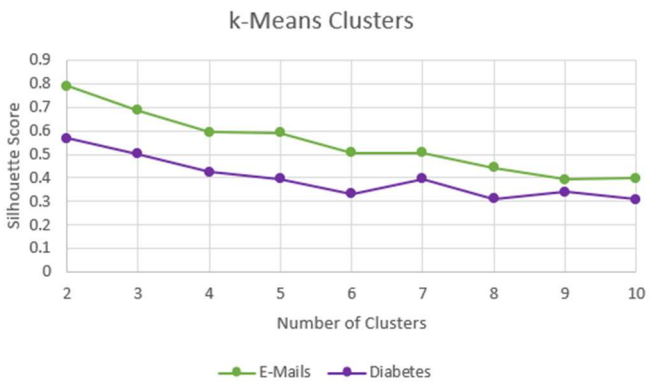
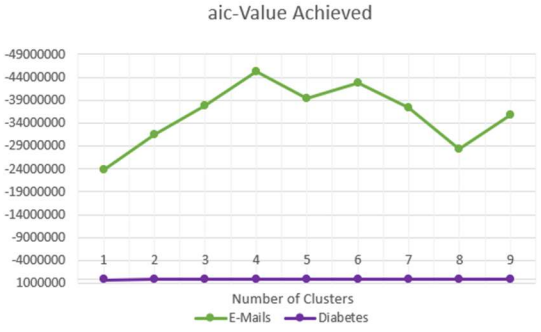


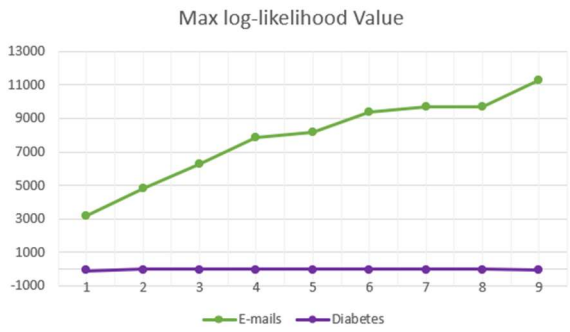
Figure 1.1 *– Silhouette score of K-Means Clustering algorithm as a function of k.*

Nonetheless, the decline is substantial enough to show a trend in the algorithm, where k = 2 represents the most distinct set of clusters for each individual dataset. As these are originally binary classification datasets, it is interesting that the optimal number of k clusters for both appears to be two. In order to further explore this observation, a count was taken from the original classifications of each point, as well as the newly created clusters. Prior to clustering, the emails dataset contains 3,672 records which are classified as non-spam, and 1,500 which are classified as spam. Additionally, the diabetes dataset is initially comprised of 500 negative classifications, and 268 which are positive. Following the K-Means grouping, the emails were reported to be split into groups of 4,803 and 369, and the diabetes set was split 603 to 165. As the clusters are indistinct in terms of their classifications, this leaves the comparison up to interpretation. Since both datasets originally contain far more negative classifications, the smaller clusters of 369 and 165 can be interpreted as 'distinct classifications', or those points which are easily characterized as positive classifications (class_val = 1). If this is the case, it can be observed that 60% of the initial diabetes-positive patients were distinctly classifiable given the data, while the other 40% are more obscure and harder to distinguish. Conversely, only 24% of the initial spam e-mails can be considered distinctly classifiable under this model, leaving the remaining 76% of spam e-mails as 'fuzzy' points which are less distinct, being grouped in by K-Means with the non-spam. Overall, though, this comparison is derived largely from intuition, and is not a direct representation of the data produced by the algorithm, but rather an observation.

Figures 1.2, 1.3 – *Each dataset distribution before and after K-Means clustering. Outer rings represent the clustered distributions, while the inner rings represent the original distribution of classification values by instance.*

On top of K-Means clustering, the next clustering algorithm to be employed is the Expectation Maximization algorithm (EM). This algorithm differs from K-Means in that every data point is assigned a *relative probability* of existing in each cluster, rather than each being assigned to one singular cluster. Thus, under this algorithm, even a data point which is located at the 'center' of one cluster, is still fractionally probable to be a member of all other clusters. This notion is referred to as 'soft clustering', and results in a model which is theoretically more tolerant to real-world variance and error. In order to account for this unique model type, two scoring formulas were employed: the '*aic'* formula, and the *average log-likelihood* formula. The *aic* formula is one which accounts for model complexity in addition to cluster-cohesion, while log-likelihood is a more simplistic metric which captures the general fit of a model given the data. Generally, lower *aic* values and higher log-likelihood values indicate a well-fit model. Upon application of Expectation Maximization to the data, two distinct outcomes were observed between each set. While the algorithm successfully partitioned the *emails.csv* dataset, with an optimal cluster count of 4, it performed abysmally on *diabetes.csv,* producing negative average log-likelihood and positive *aic* values over all cluster counts. For the emails set, the optimal cluster count appears to be 4, as this is the point of maximum aic-value. While increasing the count to 6 would boost average log-likelihood, it is not worth the trade-off in complexity, as is represented by the aic-value when cluster count is 6. Although the test data pictured below showcases the performance over a range of 9 clusters, further values were tested in order to attempt to properly utilize the algorithm on *diabetes.csv*, to no avail.





Figures 1.4, 1.5 – *aic and log-likelihood scores of Expectation Maximization, as a function of cluster size.*

Regardless of cluster count, Expectation Maximization performed poorly on this data. While there are multiple reasons as to why this could be, the structure of the diabetes data is likely to blame. It is clear that over both grouping algorithms, the emails set is better partitioned. Furthermore, following the proposition of **Figures 1.2 & 1.3**, it can be assumed that the emails dataset is more varied across records. Therefore, given the distribution-percentage outputs of Expectation Maximization, it is understandable that data with higher variance could be handled more suitably by the algorithm. However, the polar difference in results is still quite stark, and may be affected by additional factors such as the relatively low instance or feature count.

Dimensional Reduction

Following initial data clustering, each dataset underwent dimensional reduction. Briefly, dimensionality-reduction is the process of taking a dataset and reducing its' features to an equal-sized or smaller set, either by eliminating features, deriving new ones from what currently exists, or a combination of both. The goal of this is to reduce memory and calculations, while preserving as much information as possible. Within this study, three reductional algorithms were analyzed: Principal Component Analysis (PCA), Independent Component Analysis (ICA), and Random Projection (RP). While each of these algorithms shares a common goal, they all employ unique methods in order to reach it; this is evident in the results output by each technique. To begin, each dataset underwent PCA. PCA reduces feature sets by finding linear combinations of the existing feature set, and using these combinations as a new, unique set of features. These linear combinations are measured in eigenvalues, a metric used to quantify covariance or instance randomness.

Figures 2.1, 2.2 *– Eigenvalue distributions of PCA-produced feature sets (5 components).*

Pictured above are the graphical representations of the eigenvalue distributions for each dataset following PCA. Each separate shade represents its' own linear combination, displayed as a percentage of the total covariance explained by that component in a set of 5 components. Looking at the charts, we see that in both new feature sets, the vast majority of covariance is covered by the first principal component, indicating that the optimal number of linear combinations is only one for each dataset. Arguably, the second principal component of *diabetes.csv* captures some relevant information, but anything beyond this is trivial. This signifies a substantial reduction on both fronts, with emails being reduced from 3,000 features to 1, and diabetes dropping from 8 features to a maximum of 2. Subsequent to PCA testing, each dataset underwent ICA reduction; while PCA can be interpreted using the eigenvalues of each component, ICA and RP produce highly obscure results, and are thus difficult to visualize. While there are methods of measuring these algorithms' output effectiveness, for the purpose of this study, they are better judged by their applicability to re-clustering and classification. However, data was collected on the time-performance of these algorithms in an effort to visualize their standalone performance. While RP was almost constantly instantaneous across cluster counts and datasets, ICA took a notable amount of time upon each execution. Below is a measure of iterations taken to for ICA to converge, presented as a function of cluster count.

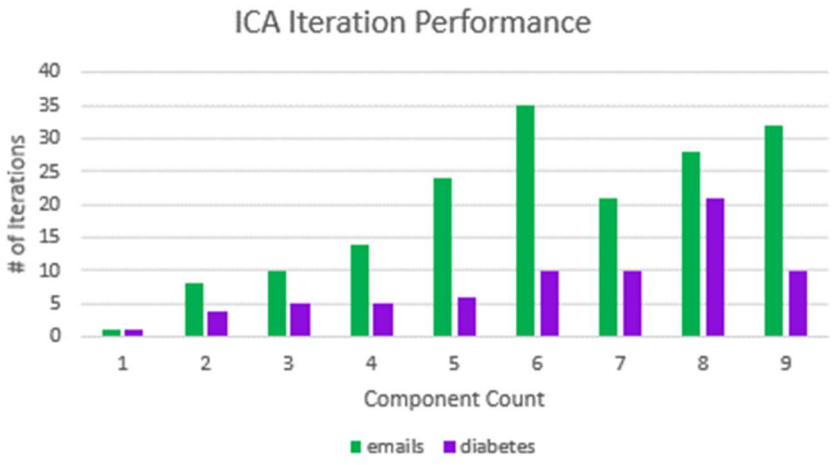


Figure 2.3 *– Iteration count of ICA, measured by increasing component count.*

One final note on dimensional reduction; while the initial decomposition solver for PCA was set to 'full' in *scikit-learn*, switching this to a randomized solver for the emails dataset improved performance, while maintaining results. This is likely due to the large size of the dataset, which explains why opting not to calculate the full decomposition would save time. All in all, the three reductional algorithms, while difficult to interpret on their own, each converged in a relatively timely manner with varying results over cluster counts.

Clustering of Reduced Feature-Sets

Once the dimensional-reduction methods were proven functional, the results of each algorithm at different component counts were then clustered by the initial grouping algorithms. This clustering differs from the original process in that rather than clustering the points using their original features, they were clustered using only the new transformed, reduced feature sets. Since it was initially discovered that the optimal cluster counts are 2 and 4 for K-Means and EM respectively, these were kept constant across testing in order to maximize results. While it is possible that these counts do not apply once the data is transformed, testing was done across higher count values, further confirming that the current counts produced optimal results upon re-clustering. Initially, K-Means clustering was performed on the PCA, ICA, and RP output sets. Of these 6 reduced sets, the one which produced the cluster-space with the highest silhouette score of .973 was the ICA-reduced emails set. This represents a highly cohesive cluster-space, signifying that the ICA-transformed email features worked incredibly well under K-Means clustering.

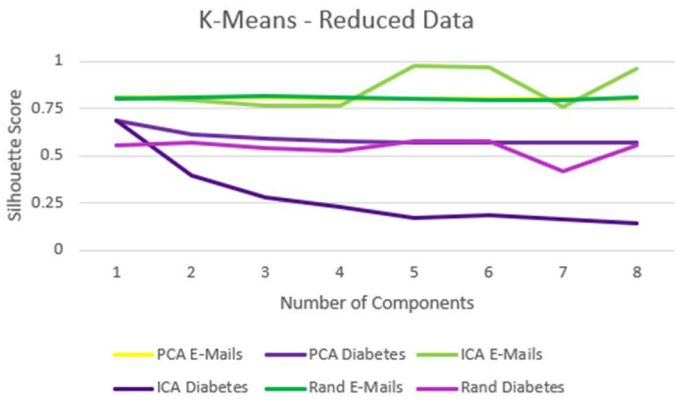


Figure 3.1 *– Silhouette scores of each reduced feature-set, passed to K-Means.*

Of note is the rise in silhouette score for these clusters at 5-6 ICA components. This demonstrates the dependence of ICA's results on the component count given. Additionally, the line trend, and outlier point at 9 components, showcase the algorithm's kurtotic output nature. Going further, it is evident that the RP and PCA email sets performed quite similarly, with their diabetes set results not too far off from one another. While it is not known why this is, it is evident that the two aforementioned algorithms both output highly similar results when clustered, which leads to the assumption that the feature sets which they produced for these datasets were also similar in composition. Additionally, is it clear that each reduced diabetes set was clustered less cohesively by K-Means than any emails set. This is most likely due to the sparse data structure, as previously discussed. Interestingly enough, ICA provided the most useful email set to K-Means, but the least useful diabetes set. This checks out, as the application of ICA can be spotty across different kinds of data, and results vary heavily based on input. Finally, it was observed that despite its' random nature, RP did not report much variance in its' emails output across component counts. This indicates that despite a rising component count, the output of RP being run on *emails.csv* remained relatively constant throughout. Following K-Means clustering of these reduced feature sets, EM was also conducted on them. While the *aic* values were much more linear than the silhouette scores, the results were not as promising. The only reduced set which was cohesively clustered by this algorithm was, again, ICA's email set. Following this observation, a conclusion can be reached that ICA provided the most useful feature set for clustering, proving effective across K-Means and EM.

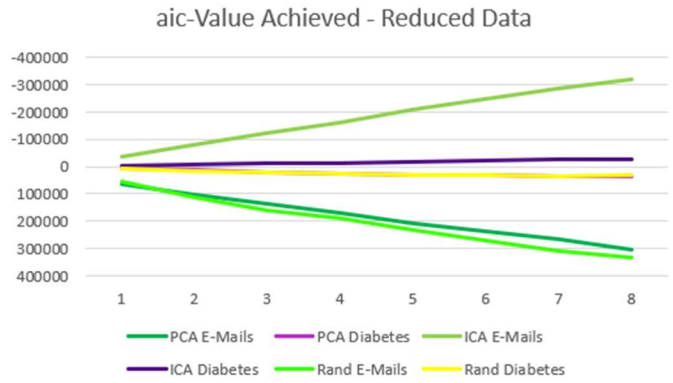


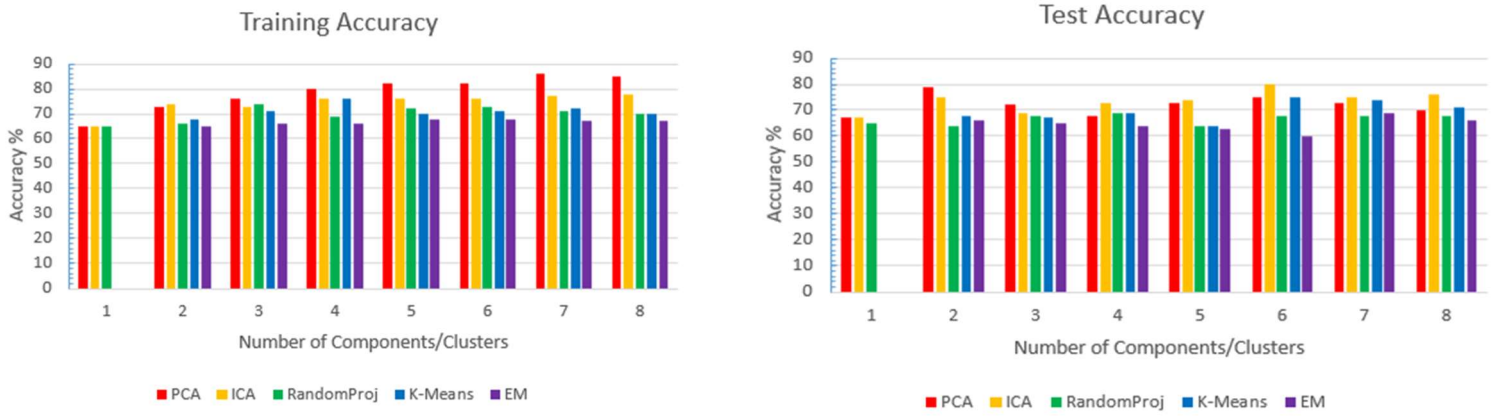
Figure 3.2 *– aic scores of reduced feature-sets following EM soft-clustering.*

Additionally, findings show that using these two datasets, ICA is perhaps the only valid dimensional reduction technique to be applied. This is likely due to the high-dimensional, sophisticated nature of the initial feature sets, and ICA's proficiency at sniffing out information from such data. However, this is certainly dependent on many factors, such as the number of iterations allowed, attempts made to improve, and the metrics which are used for scoring. Nonetheless, in this study, ICA is the clear winner when trying to reduce feature sets for the purpose of clustering.

Neural Network Training

Finally, upon the completion of all prior steps, the focus of experimentation shifted towards training a neural network. Using the aforementioned findings, the goal was to classify the *diabetes.csv* dataset, and to achieve a higher test accuracy rate than had previously been achieved with *scikit-learn*'s neural network classifier. All five of the algorithms used within this study were applied to the base *diabetes.csv* dataset, with the results being used as training and testing datasets for the classifier. Prior to the study, classification on the base dataset resulted in a test accuracy score of 74%. Employing dimensionality reduction and clustering, however, the neural network achieved test accuracy rates of up to 81%. While the classifier's parameters representing internal layer node count and learning rate were kept constant from the original classifier for consistency, these changes did not affect performance much, if at all, compared to *scikit-learn's* base implementation when faced with the reduced datasets. In order to test the effectiveness of dimensional reduction and clustering on the diabetes set, and how it affects classification, each algorithm was run a total of 3 times on the data at a range of {1-10} components ({2-10} clusters). Subsequently, each of the 3 output matrices were fed to the classifier, with the average training and test scores being recorded across the 3 runs.

Overall, PCA and ICA produced the greatest results; while PCA had the highest individual reported training rate (86%) as well as the greatest average training rate, ICA reported the greatest average test accuracy at 80%, a 6% increase from the classification score of the previous study. Additionally, ICA's feature set was not the only result to improve on *diabetes.csv*. During testing, PCA and K-Means clustering both reported high test scores of 79% and 75% respectively. These both indicate gains in accuracy from the original classifier, thus achieving the goal of the study. Given the findings of the analysis on each clustering/reduction algorithm, it comes as no surprise that ICA produced the most accurate neural network. Concise feature sets produced by the algorithm led to multiple classifiers reporting gains in accuracy from before. While three of the five classifiers reported accuracy gains following reduction, Random Projection and Expectation Maximization both fell short during the testing phase. Although Random Projection boasted training scores in the mid to high seventies, it never once cracked 70% during testing. This could be due to the random nature of the algorithm in combination with the limited implementation of the study, or other factors entirely. As for Expectation Maximization, its' training scores and test scores both consistently fell below 70%. However, this is consistent with the data in **Figures 1.4 & 1.5**, and thus makes sense when this is taken into consideration. Overall, though, the results of the study showed improvement from the original classification, all the while demonstrating the strengths and weaknesses of the various algorithms.



Figures 4.1, 4.2 *- Average training, test accuracy percentages of the Neural Network Classifier for each algorithm over 3 runs and a range of components.*