Kellen O'Connor

Assignment #3 7641

Unsupervised Learning and Dimensionality Reduction

Summary

In this paper I implement and optimize two unsupervised learning algorithms K-Means clustering, expectation maximization and four dimensionality reduction algorithms PCA, Randomized Projections, ICA, and SelectFromModel on the two datasets from assignment 1 and 2. Following my dimensionality reduction algorithms, neural networks are applied against the resulting datasets and are observed and analyzed.

Dataset Overview

The Bankruptcy Dataset:

Gathered by the Taiwan Economic Journal from 1999 to 2009 this dataset includes ninety-four metrics of company value, financial health, business performance and includes a bankruptcy flag label for companies that have filed for bankruptcy. Variables include financial metrics like cash flow per share, total asset turnover, operating gross margin, and net value growth rates. The target of this dataset is it's Bankruptcy flag, 0 for a company that has not gone bankrupt and 1 for a company that has gone bankrupt. With the presence of this bankruptcy flag it should be possible to identify the types of financial behavior or financial states of a company that is bankrupt or perhaps near bankruptcy. This Bankruptcy flag target is unbalanced however, with 220 values of 1 marking a company that has gone bankrupt and 6599 values of zero marking companies that have not yet gone bankrupt (3:100 ratio). In order to help compensate for the imbalanced nature of this dataset we will use the macro averaged F1 score for performance as it provides additional importance to underrepresented classes, in our case the Bankruptcy target class. Despite this unbalanced target column, the dataset remains interesting in that being able to identify companies or organizations about to go bankrupt could provide useful knowledge and advantages to many different parties, including the financially troubled organization itself or even different companies.

The HR Dataset:

Data put together by an organization seeking to better invest the efforts of its human resources department. This dataset was gathered to help understand what aspects about an applicant will make them more likely to accept a job offer if given one. The goal is to be able to correctly classify that types of job applicants most likely to accept and begin employment based on a series of variables that include information about education background, previous employment history, and information about the applicants existing city or location. This dataset includes variables that are non-numeric categorical, and the set is imbalanced so steps must be taken to accommodate this. 10 of the 14 features are non-numeric categorical and we use encoding in order to deal with this. The HR dataset is moderately imbalanced with it's target class 'target' containing 4777 '1.0' values, signifying a job placement, and 14,381 '0.0' values signifying no job placement. In order to accommodate this imbalance we use macro averaged F1 scores to provide greater importance to the less observed target class.

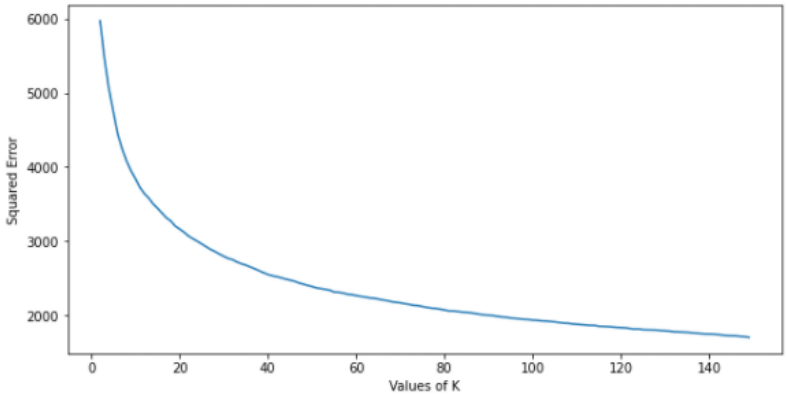
Clustering Algorithms

Bankruptcy Dataset:

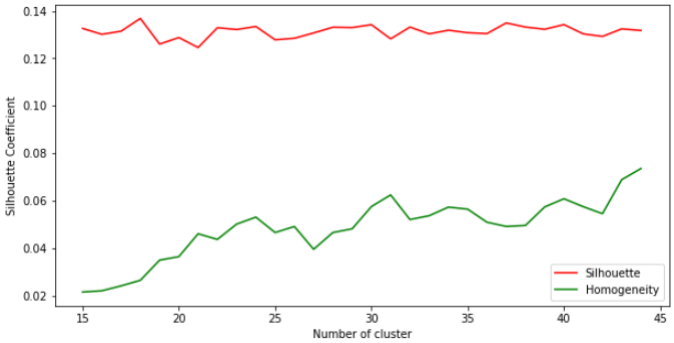
K-Means Clustering

Almost all the 90 something variables in the bankruptcy dataset are floats with decimals so because of this I chose to use Euclidian distance for the distance metric. Because of the high number of features though, all of which capturing

a wide array of financial characteristics, I was skeptical of the clustering algorithms. I used the sum of squared error to judge the value of K starting at K = 2 and ranging to K = 150, producing the figure below.



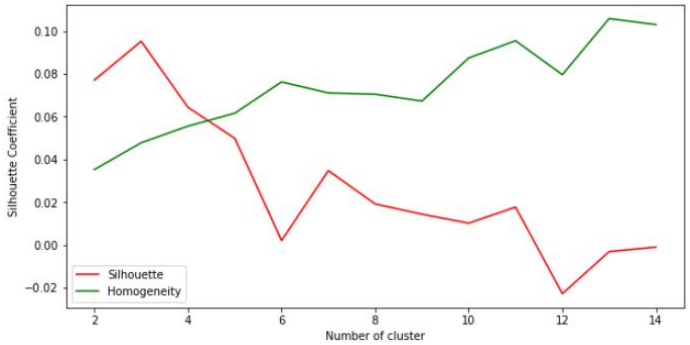
Because there is no sharp elbow in this plot, I have decided to use the silhouette coefficient in order to better determine the value of K. By seeking the optimal silhouette coefficient we can better optimize for the number of clusters. Honing in on the area of the cuver where the elbow seems to be occurring gradually, we can see optimal silhouette scores at 18, 30, and 37.



Because the Silhouette score is closest to the homogeneity score at 30 I opt to go with K = 30. In order to see if KMeans can identify the two classes represented in the Bankruptcy flag variable I set K = 2 and achieve an accuracy of 0.31, which is quite bad. I believe that this has to do with the sheer number of features provided to the algorithm maybe making the data unnecessarily complex.

Expectation Maximization

Because expectation maximization works differently than KMeans and is able to capture Gaussian Mixture variances and different shapes present in the data (**Brownlee**). I repeat the process I've used for KMeans and plot the silhouette scores against homogeneity to produce the following figure.



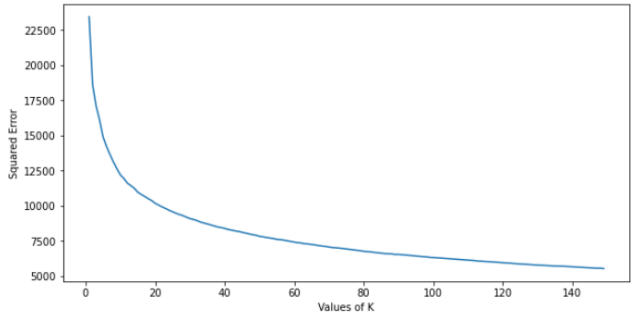
Based on this plot I opt to choose K = 4 with a silhouette score of 0.0643 and homogeneity score of 0.0555.

Setting K = 2 to create two cluster I test EM to see if it can reproduce clusters that correspond to the classes of the underlying dataset. The accuracy achieved is 0.8969 which is significantly better than the accuracy achieved by KMeans. I believe this is because EM is a little more capable of dealing with the sheer number of features compared to KMeans.

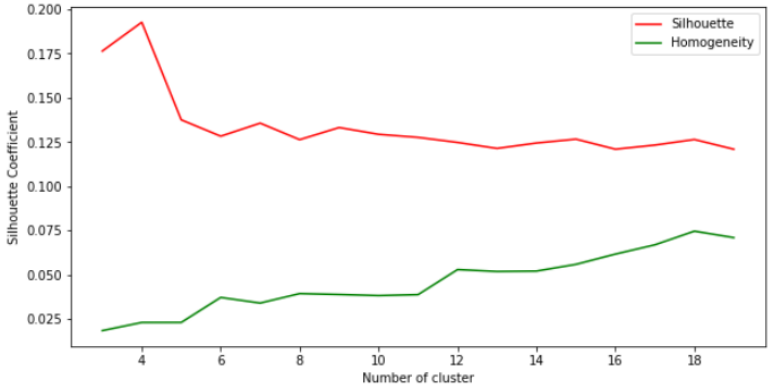
Human Resources Dataset:

K-Means Clustering

For our implementation of KMeans against the human resources dataset we use the same steps, starting by plotting the squared error against values of K. Although not entirely clear there appears to be slightly more of an elbow in the plot.



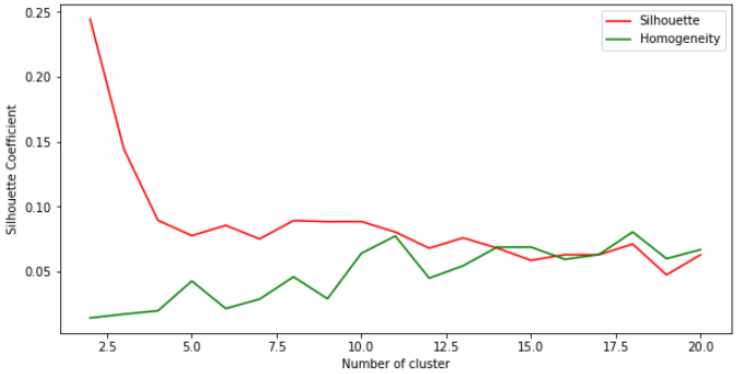
With an elbow occurring somewhere between 4 – 12 we focus on this area for the plotting of our silhouette and homogeneity scores. The plot below shows the silhouette scores plotted against homogeneity for values of K 2 through 14. With one of the peaks of the silhouette coefficient and homogeneity score at K = 7, we chose 7 as the value of K.



Just like we'd done with the bankruptcy data we test the KMeans algorithm against our target class to evaluate whether or not it can identify the two classes. With n_clusters set to 2 our implementation of the KMeans algorithm achieves an accuracy of 0.660089 which is surprising in that it's somewhat better than something entirely random.

Expectation Maximization

For our implementation of expectation maximization against the human resources data we evaluate the different values of number of clusters against homogeneity scores and silhouette scores.



Seeking a value of K that minimizes the gap between silhouette and homogeneity scores while maximizing the silhouette coefficient I opt for a value of K = 10.

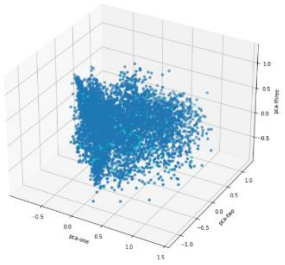
To evaluate the expectation maximization algorithm against correctly identifying the target class within the human resources data we set n_clusters = 2 and test accuracy. For the HR data and EM algorithm with n_clusters set to 2 we achieve an accuracy of 0.6601419, its similarity to KMeans shows that EM doesn't have any accuracy advantages over a simpler KMeans against the human resources data.

Dimensionality Reduction Algorithms

PCA Algorithm

Bankruptcy Data

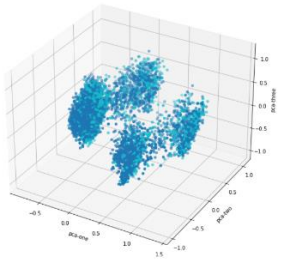
The bankruptcy dataset has 96 variables and is decently more complex than the human resources data because of this so a dimensionality reduction algorithm could reduce complexity while still capturing the needed information. I tested reconstruction error against a series of dimension values ranging from 2 to 90. PCA is successfully able to capture 0.99 of the data's explained variance at 32 dimensions and 0.92 at 12 dimensions. I opted to use 27 dimensions, capturing 0.986 of the explained variance of the bankruptcy data. When we plot the first three PCA components we observe that they are not separated in 3 dimensions.



Human Resources Data

The human resources data only has 14 variables compared to the bankruptcy data's 96 variables, making it less likely to benefit from a reduction in the number of features. This low number of initial features makes me a little

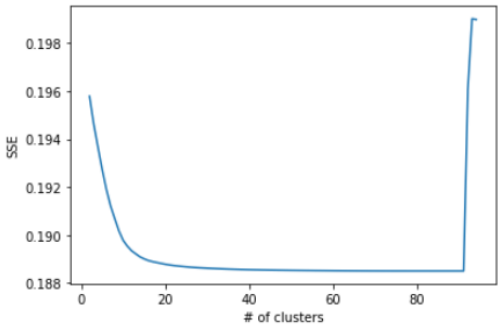
worried about the loss of information from PCA. We test PCA against new numbers of dimensions from 4 to 12 dimensions. At 10 dimensions we have an explained variance of 0.886 and at 6 dimensions we have an explained variance of 0.605 so we can see a significant loss of information quickly. I opt to use 10 dimensions and plot the first three PCA elements below.



ICA Algorithm

Bankruptcy Data

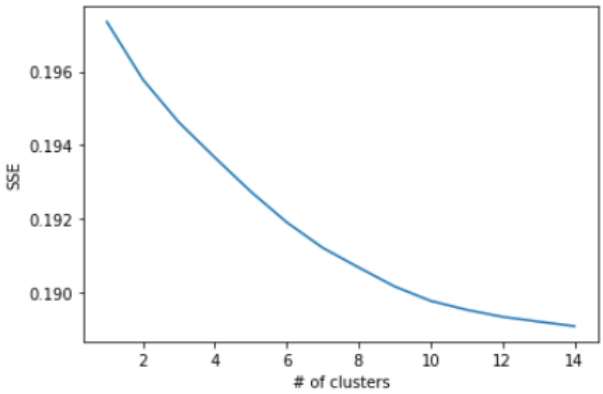
In implementing the ICA algorithm I've chosen to use reconstruction error to pick the number of dimensions that should be used. For the 95 dimensions present in the original dataset I test 1 to 95 dimensions and produce the plot below.



I opt to use 40 components from the banking data with the ICA algorithm because after 40 it plateaus and is still less than half of the dimensions from the original set.

Human Resources Data

Just as I've done with the bankruptcy data I implement the ICA algorithm against the human resources data and plot reconstruction error.

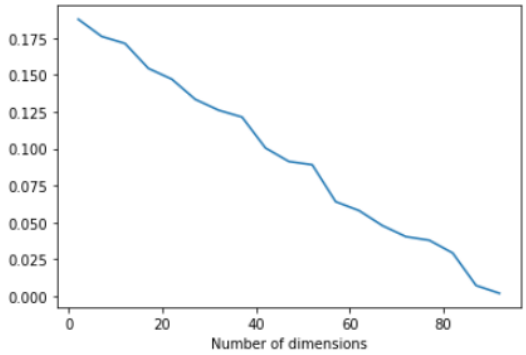


Compared to the bankruptcy data the HR data seems to more quickly gain reconstruction error with fewer features. This makes sense intuitively as there are so few variables to work with to begin with, and it matches the behavior observed from using the PCA algorithm. I chose to use 8 components as it is a decent reduction in the number of dimensions while also preventing too much information loss.

Randomized Projection

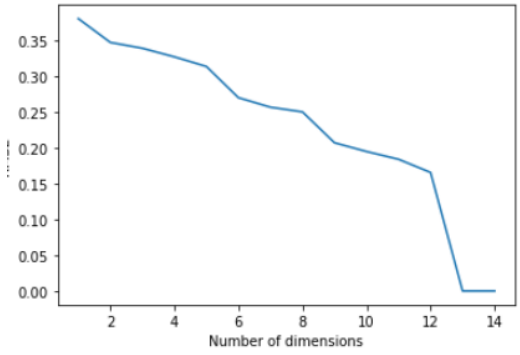
Bankruptcy Data

For the randomized projection algorithm I plot number of dimensions against error again to produce the plot below. This plot appears to be fairly linear so I opt for somewhere in the middle for my number of dimensions, selecting 45. I feel as if this a good selection because it's less than half of the original dimensions and prevents the loss of too much information.



Human Resources Data

Just like I've done for the bankruptcy dataset I plot the number of chosen dimensions for the HR dataset against reconstruction error and produce the plot below.

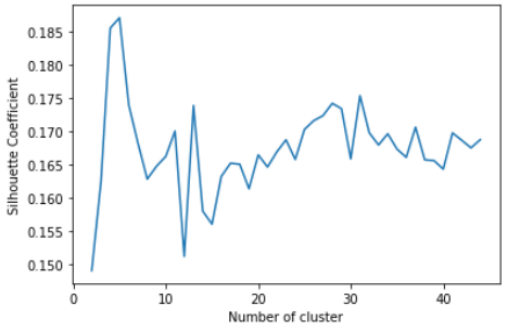


Less linear than the Bankruptcy dataset there does appear to be a few spikes and peaks depending on how many dimensions that are chosen. Because of this behavior I choose to use 9 dimensions to prevent too much information loss while also still reducing features used.

SelectFromModel

Bankruptcy Data

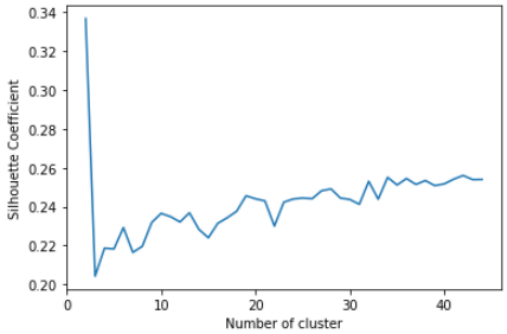
Lastly I chose to use the SelectFromModel algorithm to choose a number of dimensions similar to what we've done with the other three algorithms already discussed. To evaluate I transform the dataset and reduce to 20 components from 95 and then generate a KMeans Silhouette score against the number of clusters and produce the figure below.



This figure is interesting because at several points the silhouette score from the SelectFromModel transformation of the original data outperforms the regular dataset.

Human Resources Data

Just as I've done for the bankruptcy data I implement the SelectFromModel against the human resources data as well. Transforming the original data down to 8 dimensions from 15, we calculate the KMeans silhouette scores and then plot them against the number of clusters to produce the figure below.



Just as with the bankruptcy data I was similarly surprised to see that at multiple points the SelectFromModel transformation of the HR dataset outperforms the regular HR dataset.

<u>Dimensionality Reduction Algorithms and Clustering</u>

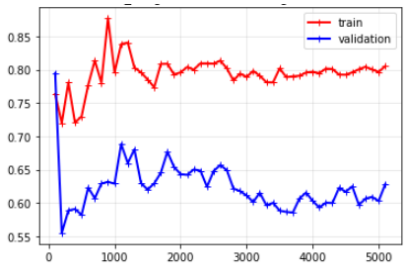
In total I have completed all 16 clustering algorithms and wanted to briefly document some interesting findings before moving on. I was particularly surprised by the degree to which we could reduce the number of dimensions used in the Bankruptcy dataset, for example down to 27 dimensions from 95 in our implementation of PCA while maintaining explained variance of 0.98 This leads me to believe that many of the variables are related to one another and that their transformation does not lead to much information loss. Furthermore, on both algorithm's I was surprised to see that several algorithms outperformed in terms of silhouette score against the original, unmodified dataset. I believe this shows how clustering can become easier after projection or the reduction of dimensions when you have many features in your dataset. In particular SelectFromModel performed best in terms of silhouette score against the original HR dataset.

<u>Neural Networks against Bankruptcy Data after Dimensionality Reduction</u>

I chose to implement the neural network against the bankruptcy data because of it's high number of features I believe that it would be able to benefit greatly from dimensionality reduction and we might be able to see some performance benefits from transforming the dataset from doing so in terms of clock time and accuracy.

NN PCA Algorithm

My implementation of the PCA algorithm projected the original dataset down to 27 dimensions but maintained 97% of the explained variance. To start I take the optimized neural network I created in assignment number 1 and run it against the transformed PCA dataset to produce the figure below.

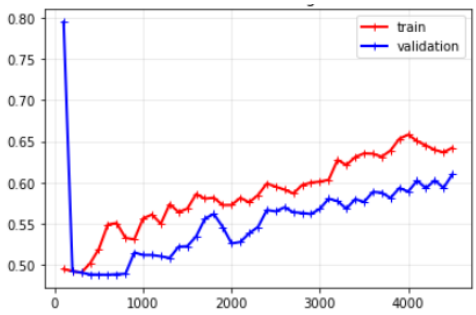


Wanting to further optimize the model I use gridsearch to try and tune the hidden network size of the model from assignment number one. What's interesting is that smaller hidden networks appear to perform well on the projected data, perhaps because the reduction in dimensions has simplified the required complexity of the neural network.

Looking at the model's performance we achieve an accuracy of 0.971 and a macro average F1 score of 0.67198. In assignment #1 we had a macro F1 score of 0.6701 so this similar and even marginally better performance is a great sign considering we've halved the dimension space our model needs.

NN ICA Algorithm

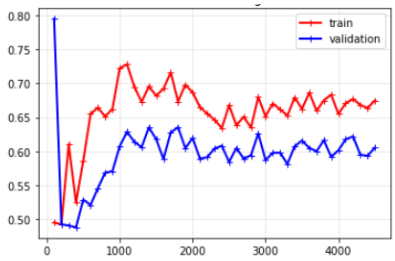
Similar to what I've done for the PCA algorithm I run our best neural network from assignment number 1 with a smaller hidden layer size against the projected ICA dataset. Our ICA dataset was been reduced to 40 dimensions down from 95. To evaluate it's performance we generate the learning curve shown below.



Against the ICA projected data our model achieves an F1 macro score of 0.5596 and an accuracy of 0.96624. Compared to both PCA and the original dataset, the ICA projection of the bankruptcy data seems to suffer more. Interestingly though, the ICA projection did seem to run moderately faster than the PCA projection and the original dataset from assignment 1.

NN Randomized Projections

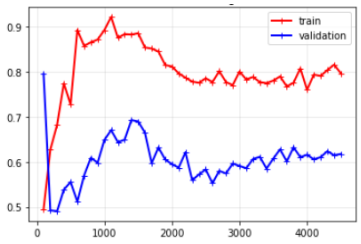
For the randomized projections algorithm we similarly run our best neural net against the randomized projection, projected bankruptcy dataset. Using randomized projections we opted to use 45 dimensions, so that is what we run the neural network against.



Our plotted learning curve shows that the Randomized Projections projection of the bankruptcy dataset achieves an F1 macro average score in between the one produced by PCA and ICA, but does worse than what the neural network was able to achieve in assignment 1.

NN SelectFromModel

Lastly we use the SelectFromModel algorithm to generate and evaluate the performance of a similar projection against our best neural net from Assignment 1, just live we have done for our three other algorithms. Our SelectFromModel algorithm is able to create a projection of the bankruptcy data that reduces our dimensions down to twenty dimensions from the original 95. We use this projection to generate and plot the learning curve below.

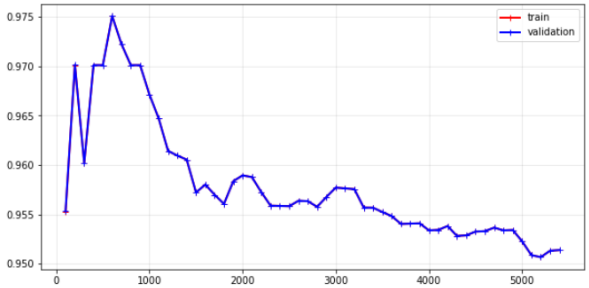


Our implementation of the SelectFromModel is capable of achieving an f1 macro average score of 0.6352 and accuracy of 0.9649. This places the SelectFromModel somewhere between PCA and Randomized Projections in terms of performance. It is worth noting however that although it isn't be best performer via macro F1 score it performs remarkably well for only consisting of 20 dimensions compared to PCA's 27.

Post Clustering Neural Network Performance

KMeans Clustering

For my use of the KMeans Clustering algorithm for this final evaluation against the neural network, I use K = 31 because at that value we saw a peak in homogeneity and small distance between the silhouette and homogeneity score in our earlier figure. Running our best neural network against the clustered data, we plot the following weighted average F1 score.



Despite the promising appearance of this score, when we examine the macro average score we see that we get a performance of 0.492 making our model little better than random chance. This implementation of the model is also very, very biased. I believe that this could be caused by the severely imbalanced nature of this dataset, with its 3:100

class imbalance ratio. Clusters are probably more likely to be drawn across other financial patterns more present in the 97% non bankrupt companies than across the patterns occurring in the 3% of companies going bankrupt.

Expectation Maximization

For our implementation of expectation maximization data against the best neural net, I unfortunately expected to experience something similar to what was observed with the KMeans Clustering output above. In order to try and accommodate the behavior observed in our KMeans attempt I tried to lower the test size percentage during cross validation, hoping the capture more observations flagged with bankruptcy in the training set, but this does not appear to have worked. Our validation curve looks different but exhibits the same behavior above.

Just as with KMeans we observe a macro average F1 score that is little better than random chance. A result that I believe comes from the bankrupt target class's key information being obscured because of the very low number of their occurrences.

Conclusion

This paper documented the implementation of two unsupervised algorithms, and four dimensionality reduction algorithms against two datasets, the bankruptcy dataset and the human resources dataset. The dimensionality reduction algorithms showed how reducing dimensional space can be a key tool to improving the computational costs of certain models, neural networks included. An interesting find was being able to reduce a dataset with almost 100 dimensions to 27 dimensions with only a 3% loss in explained variance. In general it was interesting to observe how the reduced dimensions in the projected datasets required less clock time, likely because of the simplicity of lower space state.

Citations:

1. Jason Brownlee, "A Gentle Introduction to Expectation-Maximization (EM Algorithm)," *Machine Learning Mastery* (blog), October 31, 2019, https://machinelearningmastery.com/expectation-maximization-em-algorithm/.
2. Tirthajyoti Sarkar, "Clustering Metrics Better than the Elbow-Method," Medium, September 6, 2019, https://towardsdatascience.com/clustering-metrics-better-than-the-elbow-method-6926e1f723a6.
3. Vikas K. Garg and Adam Kalai, "Supervising Unsupervised Learning," *ArXiv:1709.05262 [Cs, Stat]*, February 16, 2018, http://arxiv.org/abs/1709.05262.
4. Lipton, Elkan, and Naryanaswamy, "Optimal Thresholding of Classifiers to Maximize F1 Measure."