CS4641 Unsupervised Learning and Dimensionality Reduction Analysis

Colby Tobin
Professor: Charles Isbell
March 24, 2019

Introduction

The first problem was classifying the acceptability of a car given the characteristics provided within the datasets. The acceptability of the cars was ranked as unacceptable, acceptable, good, or very good. Each car also had a number of attributes associated with it, such as the purchase price, the maintenance level, the number of doors, the number of people it fits, the size of the luggage boot, and the safety estimation of the car. The purchase price and maintenance level of the car was categorized as low, medium, high, or very high. When processing the data, to make the training easier, I converted each of these values into a single value ranging from 1 to 4 respective of their categorization. The luggage boot size and safety estimation was categorized as low, medium, and high, so I converted each of these values into a single value ranging from 1 to 3 respective of their categorization. The number of doors and number of people categories used number representations except for their highest categorization. To account for this, I replaced the string to represent the highest classification with a number proportionally greater than the previous classification to make training easier. These attributes are all relevant because domain experts, at the time the data was collected, would describe the quality of a car based on these attributes. Regarding the data, there are 1728 instances of data with 6 attributes.

The next classification problem deals with classifying tic tac toe games as a win or a loss for “x”. The wins are represented as a positive one and a losses are represented as a negative one. The 9 attributes of this data set represent the different spaces within a tic-tac-toe endgame board, where each attribute can take on the role of “x”, “o”, or a blank space. To account for this, I manipulated the data, so that an “o” is a 0, a “x” is a 1, and a blank space is 2 for each of the 9 attributes. The data has been classified as a win or lose based on the rules of tic-tac-toe. In this data, there are 958 instances with 9 attributes for each instance of data.

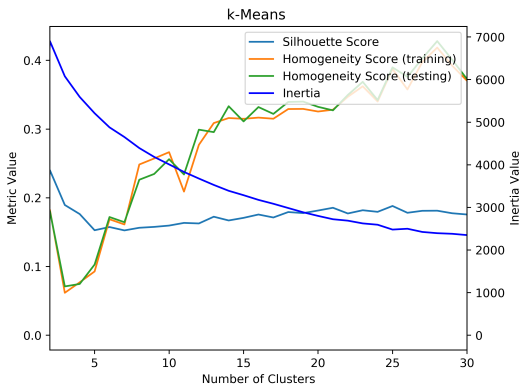
Each of these data sets are complete, which means that all attributes are present for each of the instances. In addition, these data sets also result in a single output. In order to perform training and testing of the data, I split the data into a 70% training and 30% testing set. The 70/30 split was chosen based on previous findings using the datasets I selected.

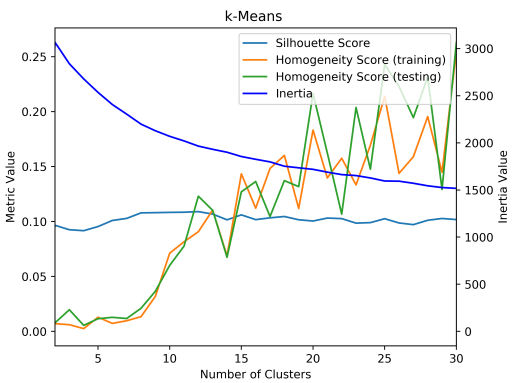
The paper has been broken down into 3 main sections. First, I will analyze the two clustering algorithms, Expectation Maximization and K-means and how that impacts the datasets. Then, I will analyze how dimensionality reduction using PCA, ICA, Randomized Projections, and k-Best actually manipulate the data. Finally, the manipulated data using clustering and dimensionality reductions will be inputs to a neural net ts to determine the new performance of the neural net.

Clustering

k-Means Clustering

This algorithm is designed so that the data is clustered into groups, so that there are *k* groups. In order to group the data, distance is used as the measurement for the grouping, and in this case, Euclidian distance is used as the distance. To determine the best number of clusters, the *k* value was varied.





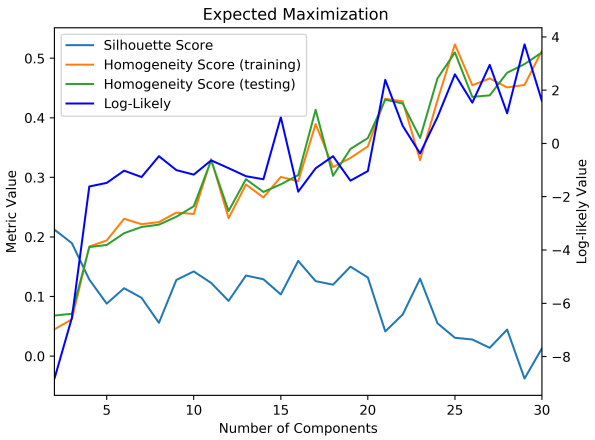
Car Quality: k-Means Tic Tac Toe: k-Means

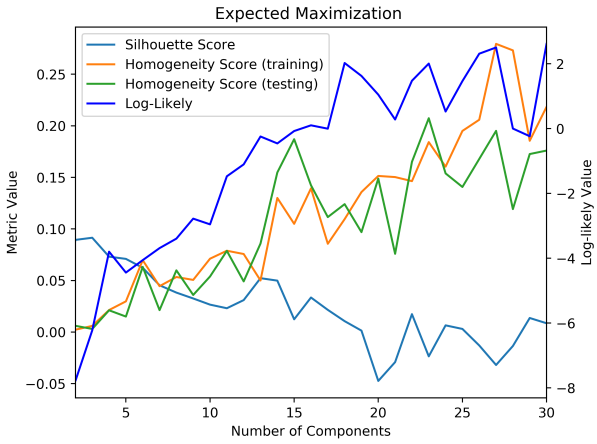
There are 3 metrics that have been presented in the graph, the silhouette score, the homogeneity score, and the inertia of the clustering. The silhouette score is a range between -1 and 1, and it represents the density of the clustering where negative values indicate that a sample has been assigned to the wrong cluster. A 0 value represents overlapping clusters. The homogeneity is a value that ranges from 0 to 1, and the homogeneity of a cluster represents the extent to which a cluster contains only members of a single class in that respective cluster. This score is not dependent on the order of the class labels. The inertia of the clusters represents the sum of squared distances to the closest cluster center using the Euclidian distance for this distance metric. The metrics are used to determine which number of clusters are best for the respective datasets.

Based on the graphs above, for the car quality dataset, it can be seen that the ideal number of clusters is around 11. This value is around the elbow of the graph for the inertia, meaning that the sum of squared difference error is not changing as much as the number of clusters are increasing. In addition to this, at 11 clusters, the homogeneity score is not increasing as much after this value of *k*, so it also shows that increasing the number of clusters also doesn't really help increase homogeneity of the clusters. Across all clusters, the silhouette stays relatively the same. With regards to the Tic-Tac-Toe dataset, the ideal number of clusters would be around 15. Similar to the car quality dataset, this appears at the elbow of the inertia curve. After 15 clusters, the overall homogeneity increases slightly, but not enough to really make a difference given the small scale shown on the left side of the graph. Like the car quality dataset, the silhouette score also does not really change as the number of clusters increase, which shows that to keep complexity of the clusters within reason and to have a small inertia value, 15 clusters is ideal for the car quality dataset. When comparing the two datasets, it can be seen that the homogeneity of the car quality dataset it much larger than the tic tac toe dataset. This means that the tic-tac-toe dataset is not really linearly separable, whereas the car quality dataset has stronger boundaries present to separate the data. Based on the silhouette scores, the clusters are grouped very close together, which makes creating distinct clusters difficult.

Expectation Maximization

This algorithm works by giving each instance of data a probability that represents the likelihood that the data belongs to a respective cluster, so the algorithm essentially maximizes the likelihood that the data belongs in their respective clusters. With each instance of data, there is extra data that is stored along with it. To determine the best clustering, the number of extra components stored with each instance of data is varied. The expectation maximization algorithm was implemented using the *GaussianMixture* from sklearn.





Car Quality: EM Tic-Tac-Toe: EM

The silhouette and homogeneity scores are the same as the k-Means clustering algorithm. The differing metric is the log-likely value. The log-likelihood metric represents the maximized probability that an instance of data will belong to a certain cluster, so a higher value for this metric means better clustering for this algorithm. Similar to before, looking at the elbow of the log-likelihood function will provide the best number of components for this algorithm that will result in the best clusters without diminishing returns for the increased complexity of more components. In the car quality dataset, the elbow is very distinct and it occurs right around 5 components. However, when taking homogeneity into consideration as well, it can be seen that having 12 components would be a better choice because the log-likely value is

higher and the homogeneity value is much higher as well. In the tic-tac-toe dataset, the elbow is less distinct, but it still occurs right around 15 components. After 15 components, the homogeneity only increases slightly, and the silhouette score doesn't change much either, which means that 15 is an ideal number of components, since it represents a balance between better and complexity.

Comparison

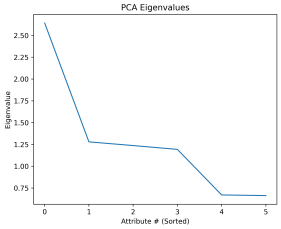
Both clustering algorithms resulted in clusters that are not homogenous, meaning that there are more than one class labeling in each cluster. In addition, both algorithms resulted in very low silhouette scores, which means that the clusters are grouped very close together demonstrating that it is very difficult to make distinct clusters for these datasets. With both datasets, there is a large number of dimensions, which is part of the reason why it is hard to cluster the points, and why the clusters are very close together. The algorithms produced clusters that were comparably "correct" for both datasets; however, the EM algorithm (60s) took a much longer time than the KM algorithm (5s). This makes sense since EM deals with complicated floating-point values and assigns vectors to each instance of data instead of just indexing the instances.

Dimensionality Reduction

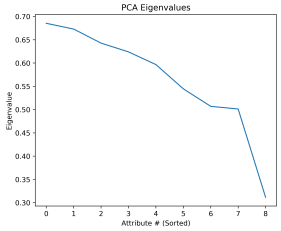
The curse of dimensionality is a reoccurring problem within machine learning that leads us to determine if the number of dimensions within a problem can be reduced. In order to demonstrate this idea, the number of dimensions are reduced with 4 different algorithms, and then run through the previous clustering algorithms.

Principle Component Analysis

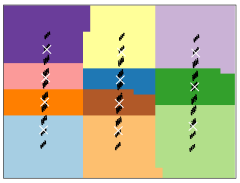
PCA works by creating a set of new axes/eigenvectors/principle components that explain the most variance in decreasing order. The size of the eigenvalue indicates how much variance exists within the data along that eigenvector or principle component. The number of dimensions for the space was set to 2, so that the clustering could be viewed in a 2D space below.



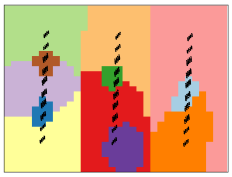
Eigenvalues: (Car Quality)



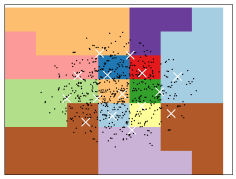
Eigenvalues: (Tic-Tac-Toe)



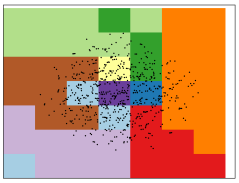
Car Quality: KM



Car Quality: EM



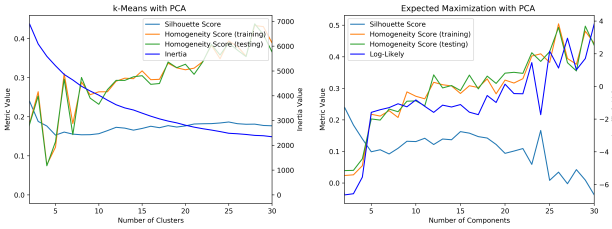
Tic-Tac-Toe: KM



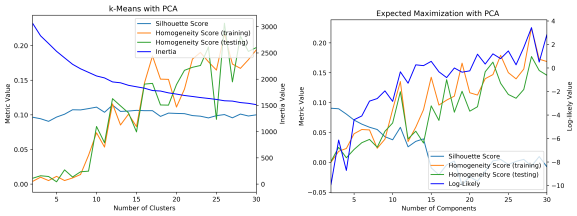
Tic-Tac-Toe: EM

Based on the first graph, it can be seen that for the car quality dataset, the eigenvalues show a sharp drop after the just the first attribute, which means that we can likely explain most of the data with just this axis. For the car quality dataset for clustering, the clustering is represented fairly densely where each cluster is very dense around its center; however, the clusters are still very close together making them hard to distinguish from each other. Based on the tic-tac-toe eigenvalue graph, it can be seen that there is not a sharp drop in eigenvalues until after the 8th attribute. This gives rise to the idea that almost all of the axes must be used to explain the data given. When looking at the clusters for the tic-tac-toe dataset, it can be seen that these clusters are not as dense as the car quality dataset, and are much harder to

determine distinct clusters due to the data not being able to be reduced as much given that almost all of the axes are required to explain the data.



Car Quality on PCA: KM and EM



Tic-Tac-Toe on PCA: KM and EM

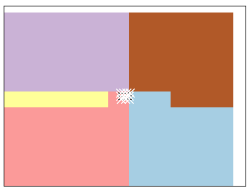
Based on the above graphs, it seems that the correct number of clusters was attained based on analyzing the respective log-likely functions and inertia functions for each dataset. Because of the dimensionality reduction, the clusters will now be in different places because they exist in different dimensional planes now. When examining the silhouette score, it shows that the clusters are slightly denser than before dimensional reduction took place. Additionally, when looking at the homogeneity of the clusters, it can be seen that it has decreased slightly as well, which makes sense because the reduced dimensionality resulted in different clusters, which explains why the homogeneity is lower, since there are not as many features to explain the data. The reconstruction error was determined by looking at the sum of squared error difference between the reduced data and the original data. For the car quality dataset, the reconstruction error reached below 10% just after 1 attribute, whereas in the tic-tac-toe dataset, the reconstruction error did not reach below the 10% threshold until after the 8th attribute, which gives rise to the idea that the tic-tac-toe dataset requires more dimensions to explain most of the data.

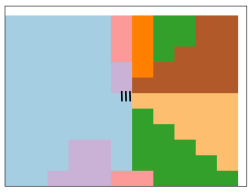
Independent Component Analysis

ICA is an algorithm that performs dimensionality reduction by analyzing the signals that reflect the data by using kurtosis measurements. To run ICA correctly, the number of dimensions for the algorithm was set to the number of attributes of each of the datasets. Below, the table shows the kurtosis measurements for each of the attributes.

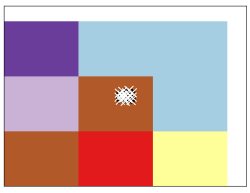
Dimension	1	2	3	4	5	6	7	8	9
Kurtosis (Car)	-1.377	-1.347	-1.307	-1.482	-1.492	-1.510	N/A	N/A	N/A
Kurtosis (Tic-Tac-Toe)	-1.156	-1.310	-1.133	-1.307	-0.958	-1.272	-1.199	-1.299	-1.216

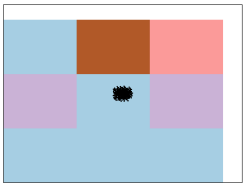
When looking at the table, both datasets present attributes that have a negative kurtosis value meaning that the distribution derived from the attribute has lighter tails and a duller peak than the normal Gaussian distribution. In the tic-tac-toe dataset, it can be seen that all of the dimensions show a similar kurtosis value, which means that this presents the idea that all of the features might be necessary in order to describe the data. Similar to the tic-tac-toe dataset, the car quality dimensions show similar kurtosis measurements; however, it can be grouped into 2 sets of dimensions, where 1 set of dimensions is further from being a Gaussian distribution. This gives insight to the idea that the data can be described with half of the features.





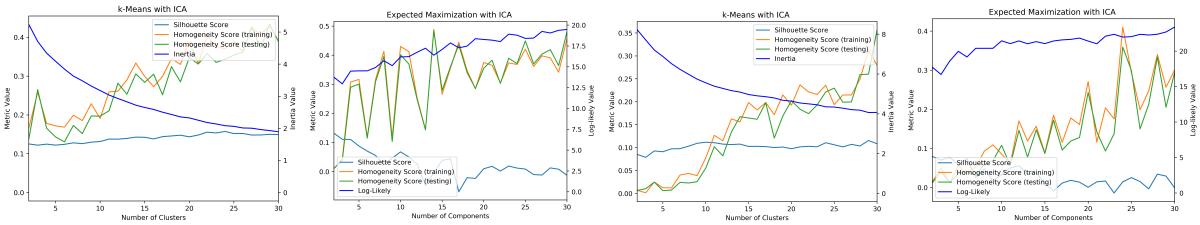
Car Quality on ICA: KM and EM





Tic-Tac-Toe on ICA: KM and EM

Because ICA is not about the projections axes, and focuses more on the linear combinations of the axes, the above clustering a graphs are not as relevant. This makes sense considering how the above graphs are very hard to distinguish actual clusters visually. With the optimal number of dimensions, ICA was clustered to show the following graphs.

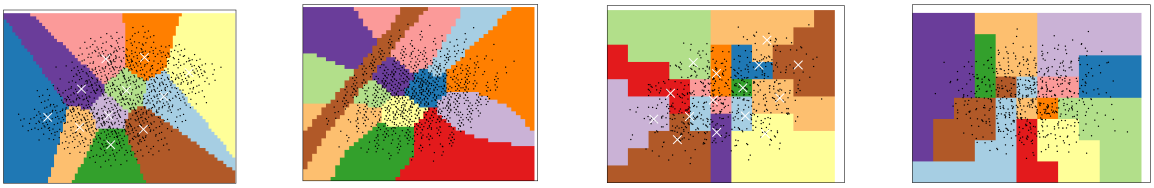


Car Quality on ICA: KM and EM Tic-Tac-Toe on ICA: KM and EM

Based on the above graphs, for the *k*-means algorithm on both datasets, it can be seen that the correct number of clusters was achieved based on analyzing the inertia functions on these graphs by looking at the elbow. However, when looking EM graphs, it can be seen that the log-likely function does not have a similar L-shape as the previous graphs. Overall, it has a linearly increasing graph, which makes it difficult to decipher where the best number of components is for this clustering algorithm for both datasets. When looking at the graphs, it can be seen that the silhouette score is much closer to 0 in all cases. This means that the clusters after this dimensionality reduction are starting to overlap with each other making them harder to distinguish. This can also be seen in the colored cluster graphs above. Compared to the original homogeneity of the clustering algorithms, they are relatively similar, but slightly higher, which shows that the composition of the clusters is purer compared to before after dimensionality reduction.

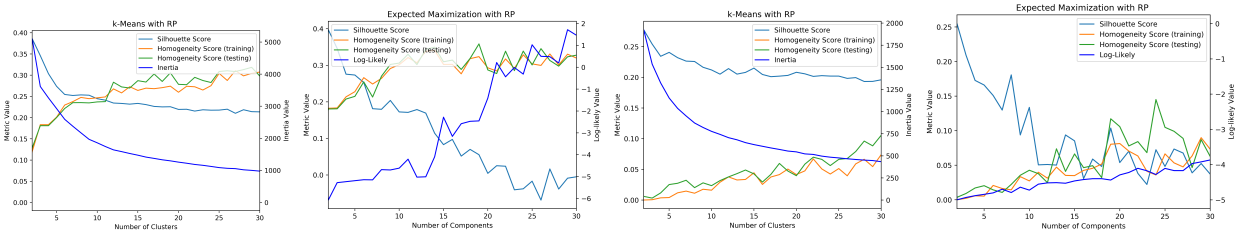
Randomized Projections

The RP algorithm generates a random matrix using a Gaussian random matrix in the hopes that this results in a good transformation of the data. It is different from previous algorithms as it does not maximize variance or minimize error. Because of this, RP works much faster than previous algorithms; however, it results in a different transformation of data each time it is run because it is done randomly. In order to visualize the clusters, 2 dimensions were used to view the clustering in the below graphs.



Car Quality on RP: KM and EM Clusters Tic-Tac-Toe on RP: KM and EM Clusters

Based on the above graphs, it can be seen that this run of the RP algorithm results in a good transformation of data as the clusters presented are reasonable. When looking at previous algorithms, these clusters presented above are less densely centered on the cluster centers and are more spread out.



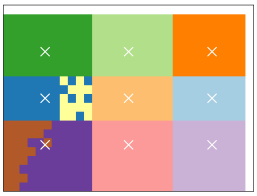
Car Quality on RP: KM and EM Tic-Tac-Toe on RP: KM and EM

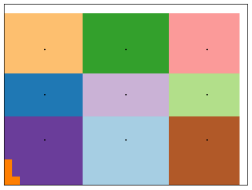
Based on the above *k*-means graphs, it can be seen that the correct number of clusters was achieved again after performing randomized projections. This can be seen based on looking at the elbow of the inertia curves. When looking at the EM graphs, the log-likely curves follow an overall increasing linear pattern rather than an L shaped curve making it hard to distinguish the ideal number of components for this clustering algorithm. When comparing the homogeneity score to before RP, it can be seen that the homogeneity is much lower after perform dimensionality reduction on this data using RP. This could likely be due to the random nature of the algorithm. In addition, the silhouette score is much higher

after performing RP on the data, which means that this clustering configuration is appropriate for the data. When looking at the reconstruction error of the RP algorithm, for the car quality dataset, the sum of squared error drops below 10% after running through 3 components. For the tic-tac-toe dataset, the sum of squared error drops sharply to around 2% after running through 4 components. This is a very low value for this algorithm. However, when looking at the overall nature of the algorithm, because it is randomized, the algorithm would produce different results every time and would transform the data differently with each run, so these clusters and reconstruction errors would change.

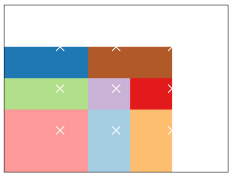
k-Best

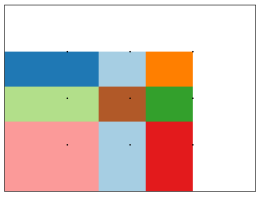
This algorithm scores each of the features using a defined scoring function. It will then select the *k*-best features and reduce the data by removing worse features. In the case of this problem, the scoring function used was the mutual information, which measures the mutual information between two variables. This was chosen due to better domain knowledge on the subject and based on the comparison to an alternative scoring function of the F-value.





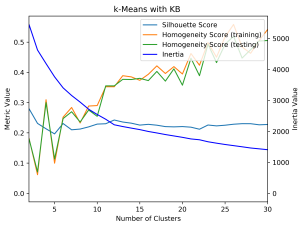
Car Quality on KB: KM and EM clusters

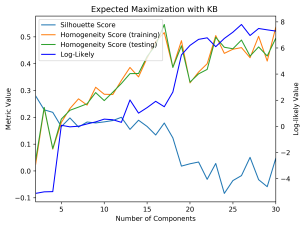




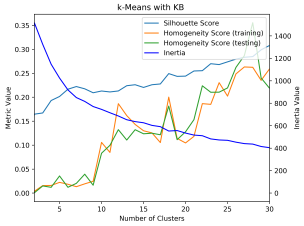
Tic-Tac-Toe on KB: KM and EM Clusters

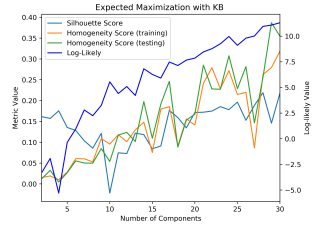
It is clear from the clustering that each of the clusters are very tightly centered around their respective cluster center when using mutual information as the scoring function.





Car Quality on KB: KM and EM





Tic-Tac-Toe on KB: KM and EM

Similar to previous algorithms, this produced an optimal number of clusters and components that was the same as the original KM and EM algorithms before dimensionality reduction. For the *k*-means clustering, the silhouette score is much higher than before dimensionality reduction giving rise to the idea that the data is more densely clustered, which is evident in the cluster graphs above. The EM silhouette score is very similar to its prior score before dimensionality reduction. With regards to the homogeneity scores of each of the graphs, they are very comparable to the original homogeneity score, which means that the composition of each of the clusters has not really changed much after being reduced.

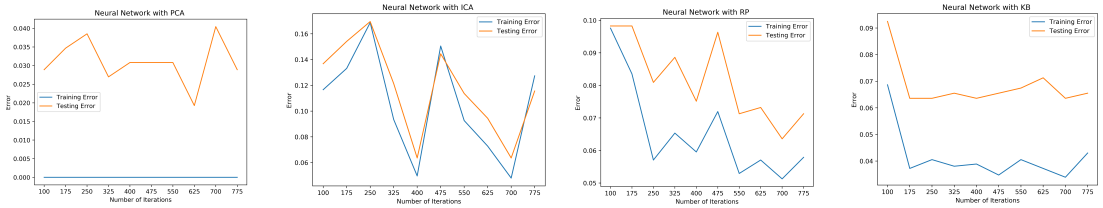
The dimensionality reduction algorithms had different effects on the different datasets. With the car quality dataset, it seemed to help the clustering by reducing the effect of the high number of dimensions, but with the tic-tac-toe dataset, it did not seem to help the clustering as much. The difference in effect on the datasets will be discussed later on in the paper. With regards to the wall-time execution time, RP executed the fastest, and then KB was second fastest, and this is due to the nature of the algorithm. ICA and PCA performed in a very similar amount of time, but they were much slower than the other 2 algorithms. This is solely based on the algorithms themselves, not the data. While PCA took a longer period of time, the reduced data it produced resulted in better clustering that resulted in similar homogeneity to before, but it resulted in denser clusters that were more distinct from the original clusters. This provides the idea that PCA would be good at finding patterns in unlabeled data.

Neural Network Training

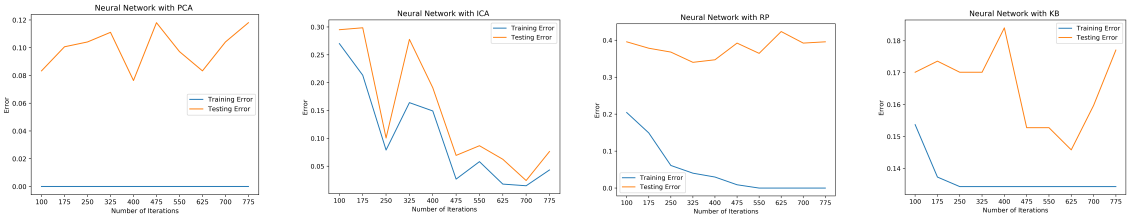
To compare the effects of these above algorithms on the actual performance of a neural network, the inputs will be changed for the neural networks. Similar to assignment 1 and 2, the accuracy of the neural network will be measured after determining the error in the training and test sets after training the neural network on the training set. In order to see how convergence changes with the reduced data, the independent variable will be the number of training iterations. The number of hidden layers was set to a constant value that was determined to be ideal in assignment 1 (165 hidden layers for tic-tac-toe and 35 hidden layers for car quality).

Dimensionality Reduction

For the following graphs, the data was run through the dimensionality reduction algorithms, and then the reduced datasets were then used as inputs for the neural network.



Car Quality after Dimensionality Reduction



Tic-Tac-Toe after Dimensionality Reduction

On the raw data, the neural net had a 0% training error and 2% testing error on the car quality dataset. When looking at all of the graphs of the neural networks for the car quality data, they all performed worse on the test set than the raw data. When thinking about the data itself, this makes sense. In the car quality dataset, there is an instance of data for every possible combination of attributes, which means that this dataset represents all possible inputs for this data. By removing features from this dataset, the algorithms are not removing noise or unnecessary features, but instead, they are removing a certain set of input combinations from the dataset, which explains the worse performance. PCA performed the best out of the dimensionality reduction algorithms only performing 1% worse than the raw data. KB performed the next best performing 4.5% worse on the testing set than the raw data. RP performed very similarly to KB, but performed slightly worse at 5% worse on the testing set than the raw data. ICA performed the worst of the algorithms performing at 12% error on the testing set. When looking at the algorithms themselves, PCA seemed to converge the quickest to a low error rate that was comparable to the raw data; however, there were large fluctuations of error rate as the number of iterations changed. These fluctuations were consistent with all of the algorithms except for KB, which seemed to level off after 175 iterations.

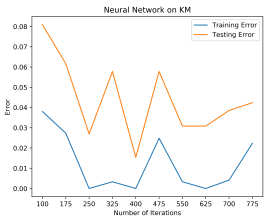
In the tic-tac-toe dataset, the baseline error rate was 0% for the training error and 9% for the testing error. On this problem, ICA performed the best as it had an error rate around 7% for the testing set. RP performed the worst of all of the algorithms at 40% error and this is likely due to the fact that there is no domain knowledge involved with this reduction algorithm. KB performed the second worst at 15% error and this is likely due to the nature of the problem. With tic-tac-toe, determining the winner of the game involves examining the entire game board, so every attribute has equal importance, so KB would not make sense as there is not feature that is truly “better” than another. ICA performed the second best performing with 12% error. Similar to the car quality problem, the tic-tac-toe problem truly does need all of its features to determine the end result, which is a likely reason for why the dimensionality reduction algorithms performed worse than the raw data for the tic-tac-toe problem as well.

For both of these datasets, it appears that they already consisted of the minimum number of features required to classify the problem correctly given that the dimensionality reduction algorithms did not perform as well on the data. This can essentially be inferred from the domain knowledge of the problem. As previously mentioned for the car quality

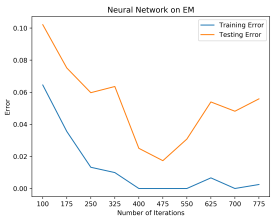
data, the dataset represents every possible combination of inputs, so removing a feature from this dataset would reduce the number of distinct combinations, which would ultimately hurt performance. With regards to tic-tac-toe data, the winner of a tic-tac-toe game can only be determined with a full game board present, which means that removing a feature would remove a space on the game board, which would result in misclassification of the problem. When comparing the training time, the neural net with raw data took a longer time to train, and this was due to the fact that the neural nets using the reduced data did not have as many features to train on, so they were able to train faster.

Clustering

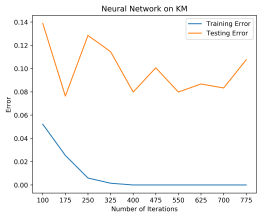
For this part of the paper, the data was run through the different clustering algorithms, and then the output of the algorithms were added to the data as metadata. This increased the dimensions of the data by 1. This new data was then used as input for the neural networks and compared to the baseline neural network.



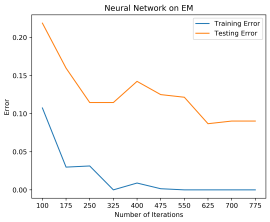
Car Quality: KM



Car Quality: EM



Tic-Tac-Toe: KM



Tic-Tac-Toe: EM

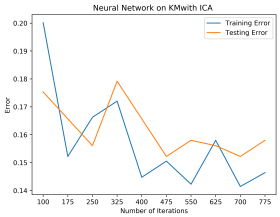
For the car quality dataset, adding the additional metadata resulted in a fairly similar performance; however, the clustering algorithms resulted in a slightly worse performance having a 4% error rate on KM and a 6% error rate on EM. With this in mind, this shows that adding the extra metadata did not truly improve the overall performance of the neural networks for the car quality dataset.

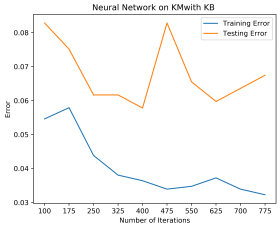
For the tic-tac-toe data, the additional metadata resulted in two completely different results for the different algorithms. For the KM clustering, this resulted in a better performance as it performed with 0% training error and 8% testing error. In addition to this, the neural network was able to converge to 8% testing error quicker than the raw data was able to converge to 9% testing error, which shows that this improved the performance of the algorithm. However, on the EM algorithm, with the additional information, the neural network performed worse than the raw data by about 1%-2% more error.

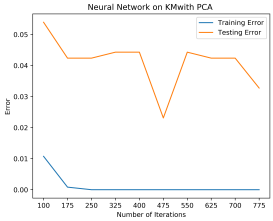
Based on these graphs, it can be seen that clustering the data did not truly improve the overall performance of the neural networks. Because of the additional information added, these neural networks took a slightly longer time to train because of the increased dimensionality. When combining the increased training time along with the overall worsened performance of the neural networks after clustering, this shows that clustering might not be truly beneficial for this dataset.

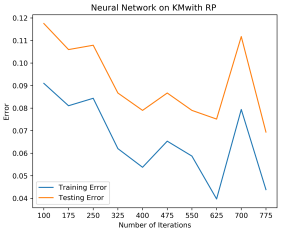
Clustering after Dimensionality Reduction

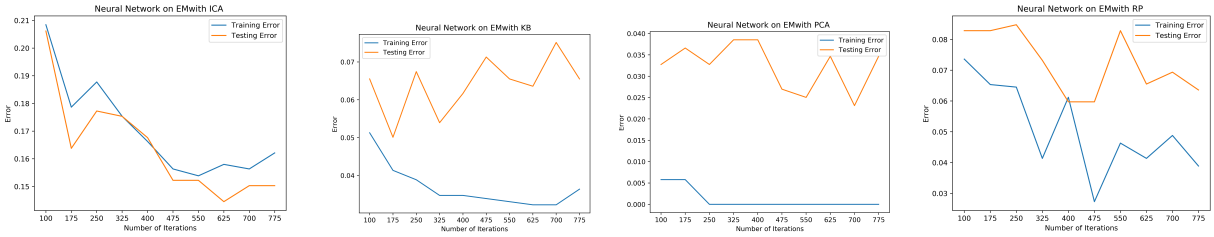
For this part of the paper, data was run through the dimensionality algorithms. The output of these algorithms was used as input for the clustering algorithms. The output of the clustering algorithms was then added to the reduced data as metadata. This new data was then used as input for the neural networks and tested against the raw data neural net.



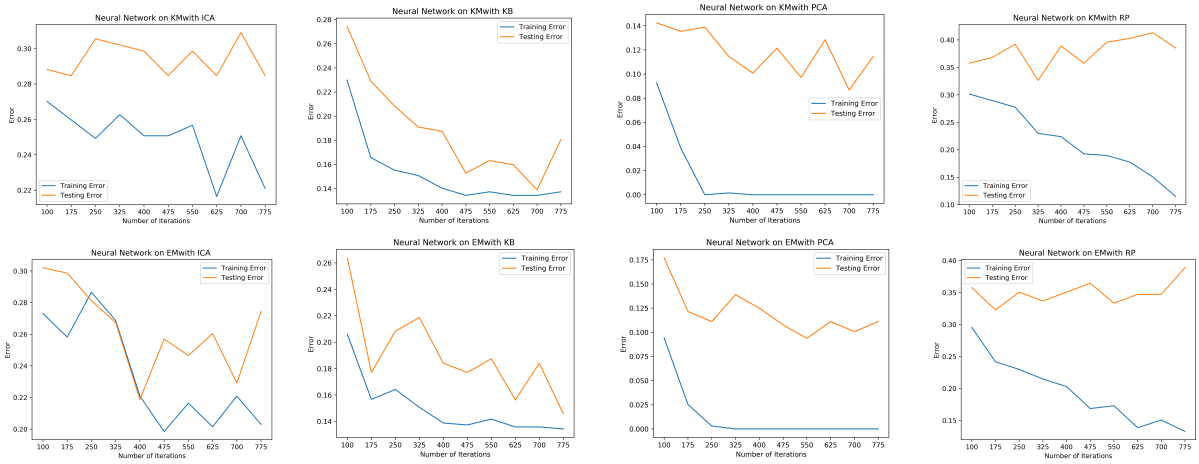








Car Quality Clustering after Dimensionality Reduction



Tic-Tac-Toe Clustering after Dimensionality Reduction

With regards to the car quality dataset, when looking at the graphs, it looks as though the different clustering algorithms on the dimensionality reduction algorithms do not impact the performance of the neural networks, meaning that for example, ICA performs similarly on KM and on EM. Each of the dimensionality reduction algorithms performs relatively similar across each clustering algorithm, and perform similarly to when only dimensionality reduction was performed on the data. This could likely be due to the fact that the dimensionality reduction algorithm helped align the data better than before, so the clustering did not truly impact or change the data much. All of these clustering/dimensionality reduction algorithms performed worse than the raw data where the raw data had 2% testing error. PCA on either clustering algorithm performed the best out of the above algorithms with around 3.5% error on the test set. KB performed the next best out of the algorithms at around 6.5% error on the test set, and RP performed slightly worse than KB at 7%-7.5% error. The algorithm that performed the worst on the data was ICA across both clustering algorithms at around 15-16% error. In all instances, clustering the data after running it through dimensionality reduction actually worsened the performance of the neural network. This is likely due to the fact that the dimensionality reduction removed attributes that truly did matter to the data, so the clustering algorithms were clustering the data without necessary information, which resulted in a worse performance.

For the tic-tac-toe dataset, the graphs follow a similar pattern to the car quality dataset where the performance is not really dependent on the clustering algorithms, but depend on the dimensionality reduction algorithms. All of these algorithms performed worse than the raw data neural net because it had a 9% test error. PCA performed the best having an error rate with around 9.5%-10% test error. The rest of the algorithms performed much worse, where KB performed the next best with an error around 15%. ICA performed the next best with an error around 26-28% error on the test set for the KM and EM clustering algorithms. RP performed the worst having around a 35% test error rate on the dataset. RP likely performed the worst due to the fact that it was reducing the dimensionality randomly with no domain knowledge. This means that if it were run again, it could result in a completely different performance with completely different data and different clustering. Like the car quality dataset, applying clustering to the dimensionality reduction algorithm actually worsened the performance of the neural network. This could likely be due to the fact that dimensionality reduction removed necessary attributes, so the clustering algorithms did not have necessary information to actually form correct clusters for the data, which could be a reason for the worse performance.

Across both datasets, the training time stayed the same compared to the raw data. This likely stems from the fact that the data was reduced and then metadata was added back to the data, so it was about the same dimensionality as before. Based on the results for the neural networks, overall, it seems as though applying these clustering and dimensionality reduction algorithms worsens performance with every new algorithm that is applied to the data.

Conclusion

This set of experiments was designed to see how clustering and dimensionality impact the data and the performance of a neural network on the data. With regards to clustering, KM performed much faster than EM overall, but they both ended up with similar results, as seen in the most recent sections. With the data, the silhouette and homogeneity scores show that the clusters are still fairly overlapping and are not very pure after performing the algorithms. With regards to the dimensionality reduction algorithms, RP performed the fastest, then KB, PCA, and the slowest was ICA. Based on the neural nets that were analyzed in the last section, it can be seen that clustering and dimensionality reduction might not be best suited for these data. The neural networks all performed worse than the raw data, and this likely stems from the nature of the problems. The tic-tac-toe problem determines the winner of a tic-tac-toe game given a board configuration. With that in mind, every attribute is equally important in determining the winner of the game. With this in mind, dimensionality reduction would only worsen a learners' ability to correctly classify a winner. The car quality problem is represented by data that has every possible combination of inputs for the dataset. Reducing attributes from this dataset could result in losing necessary combinations of data that could help the learner correctly classify attributes. In this problem, some attributes are clearly more important than others; however, the combination of attributes is the important part, and each attribute does have importance. Because of the varying importance, the dimensionality reduction in this problem does slightly better, but it is still not better than raw data.

Given more computation time, I would like to use a different dataset for an analysis as well. With my datasets, I was not able to see any improvements in the neural network algorithms, and ultimately, these datasets were not ideal for dimensionality reduction. With this in mind, a different dataset could help demonstrate the benefits and tradeoffs for each of the different clustering and dimensionality reduction algorithms. In addition, I would like to compare more scoring functions for the k-Best reduction algorithm, so it really does result in the best selection of features.

References

Algorithms Source: scikit-learn + additional documentation
Car Quality Dataset: https://archive.ics.uci.edu/ml/datasets/car+evaluation
Tic-Tac-Toe Dataset: https://archive.ics.uci.edu/ml/datasets/Tic-Tac-Toe+Endgame
GitHub URL: https://github.com/tobincolby/CS4641-UnsupervisedLearning