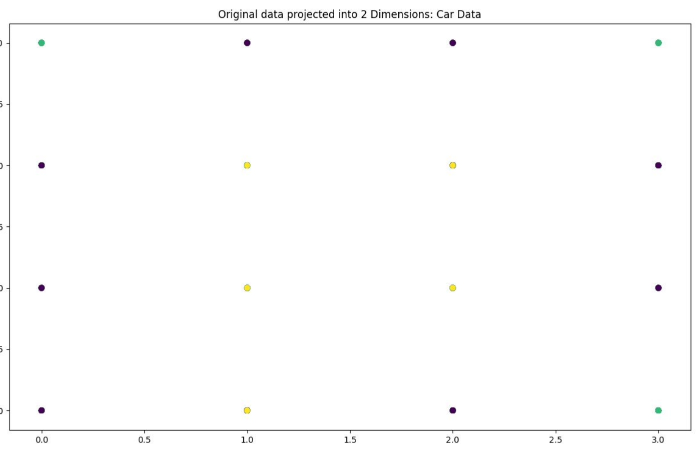
Jacob Meyers
Assignment 3 - Unsupervised Learning and Dimensionality Reduction
CS 4641
24 March 2019

Introduction

In this assignment, I explored two clustering algorithms - K-Means and Expectation Maximization - and four dimensionality reduction algorithms - Principal Component Analysis, Independent Component Analysis, Randomized Projections, and the Select K Best Features feature selection algorithm. I explored these algorithms' performance on two datasets: red wine quality and car quality. These are the same two datasets I explored with supervised learning. To test these algorithms, I performed five different experiments, each using some combination of the clustering algorithms, the dimensionality reduction algorithms, and a neural net learner.

The Data

The first dataset is the car dataset. It consists of 6 discrete features and one of four possible labels (which were originally categorical but encoded into discrete integral values for the purpose of this assignment). Even though this data is in a higher dimension than we typically plot, we can get a sense of the data's spread by effectively projecting the data into two and three dimensions by plotting the first two and three features respectively. The results are below, with the data points colored according to their final label.



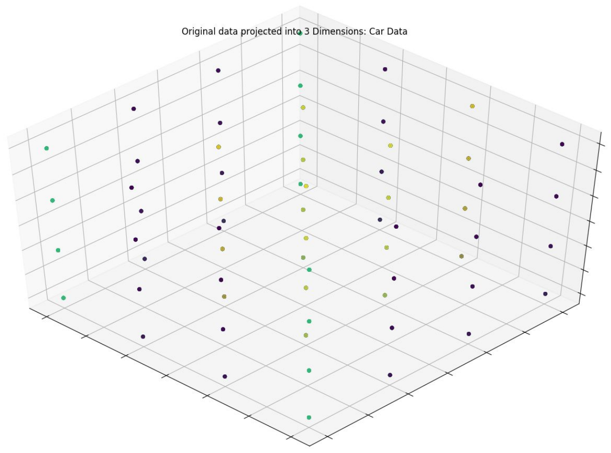
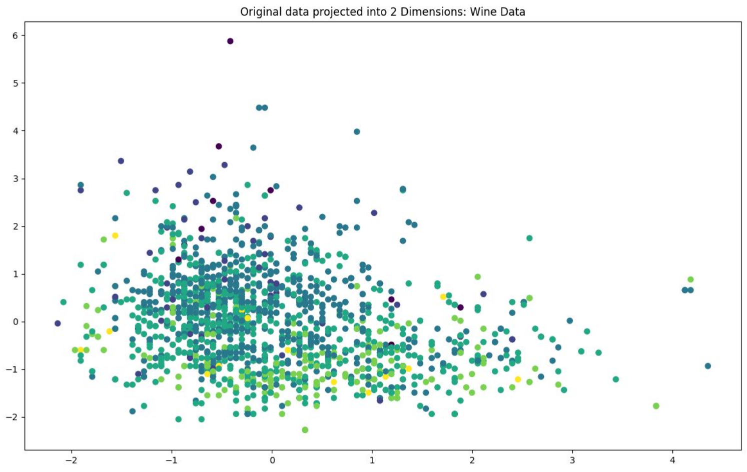


Figure 1. Original car data visualized, projected down into two and three dimensions by plotting the first two and three features.

The result is a very sparse plot. This is because of the features being discrete, and each feature having a low number (around 5) of possible values. So, despite there being around 1700 samples, it looks like there are far fewer - there are going to be a lot of examples "stacked" on top of each other, since it's much more likely that two examples have the exact same value for all features than in a dataset with continuous features. I foresaw this becoming a problem with regards to performance of the clustering algorithms, as it seemed to me intuitively that it would be difficult to find clusters that label this data correctly due to how "stacked" the instances would be and the potential of two instances with the same features having different labels. I was also interested to see what the PCA and ICA projections would look like, since this data looked to me like it formed relatively orthogonal and spread out groups to start with.

The second dataset is the red wine quality dataset. It contains 11 continuous features and a discrete label which is the rating of the wine, with a range of 3 to 8. I also projected this data into 2D and 3D space by plotting the first two and three features. The results are below.



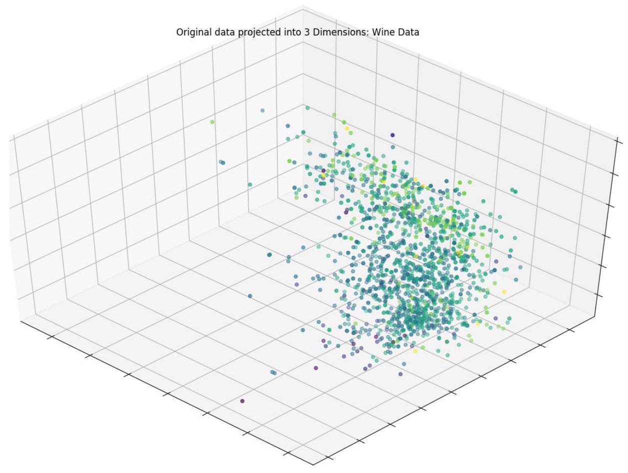
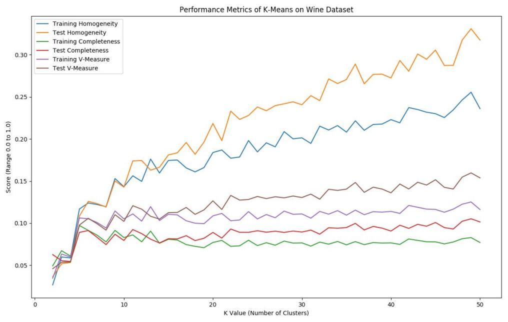


Figure 2. Original wine data visualized, projected down into two and three dimensions by plotting the first two and three features.

One thing to notice immediately is that two of the classes are much more common than the rest. The rare-er classes also don’t seem to be grouped together. However, the more common classes seem to be separated fairly well - the light green and dark green class seem to form two rough groups in 3D. This means the clusters chosen by the clustering algorithms could potentially perform well at classifying the more common class, but poorly at classifying the more rare classes. This was also the general trend with the supervised learning algorithms on this data, so I was expecting a result similar to this leading in to the experiments.

<u>Experiment One - Clustering Algorithms</u>

The first experiment I performed was to simply run the two clustering algorithms - K-means and Expectation Maximization - and test their performance. I measured performance by splitting data into training data and testing data, and for each set generating predicted clusters for the Xs. I compared these predicted clusters with the ground truth y values and generated three scores - completeness, homogeneity, and v-measure. A clustering result satisfies completeness if all of the members of a given class are elements of the same cluster. A clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class. V-measure is the harmonic mean between completeness and homogeneity. All of these metrics range from 0.0 to 1.0 inclusive, with larger values being desirable. I tested K-Means with number of clusters ranging from 2 to 50 inclusive. I tested Expectation Maximization with sklearn’s GaussianMixture object, with number of components ranging from 2 to 50 inclusive. The results are below.



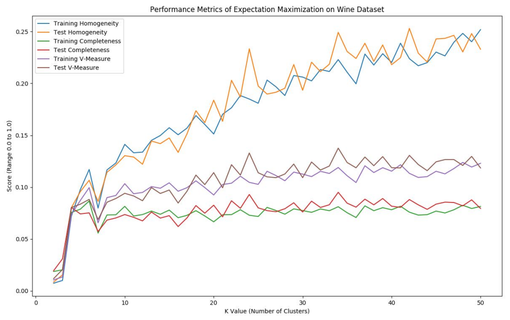
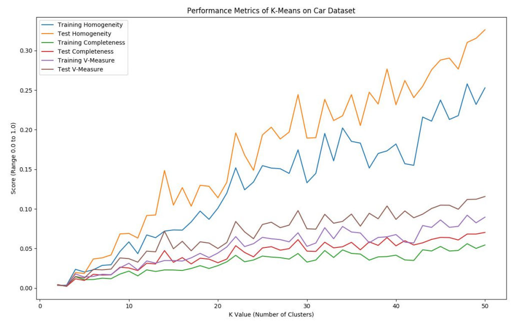


Figure 3. Results of K-Means and Expectation Maximization on the wine dataset.



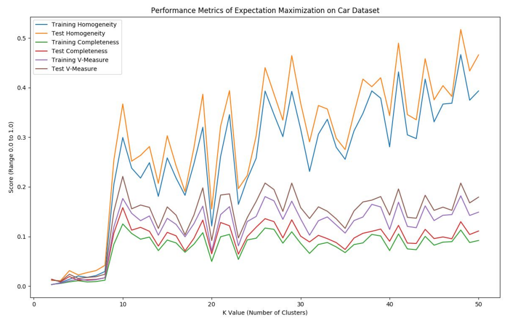


Figure 4. Results of K-Means and Expectation Maximization on the car dataset.

The first thing that jumped out to me was how poorly both Expectation Maximization and K-Means performed on the wine dataset. No metric performed better than slightly above 0.3. The best performing metric was homogeneity, which steadily increased as k increased. This is to be expected, however, as the clusters will become smaller as the number of clusters increase, which will make it easier to for a single cluster to capture a small number of members of the same class. Given how mixed up the original data is with regards to class labels, it will be difficult to create homogeneous clusters with large clusters. The completeness score, which is important in determining the accuracy in a classification problem such as this, is abysmal, barely cracking 0.10. Clearly, these clusters do not line up well with the labels. As a quick test, I visualized the clusters generated by K-Means with k = 6, the number of possible labels in the wine data. The result is below, in Figure 5.

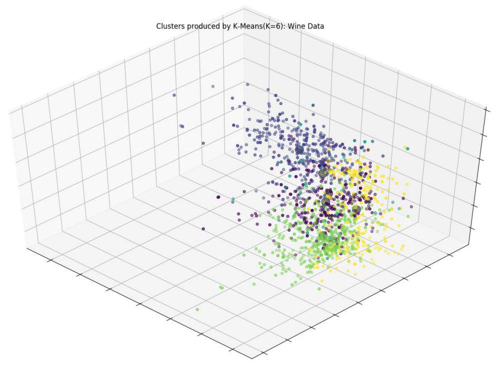


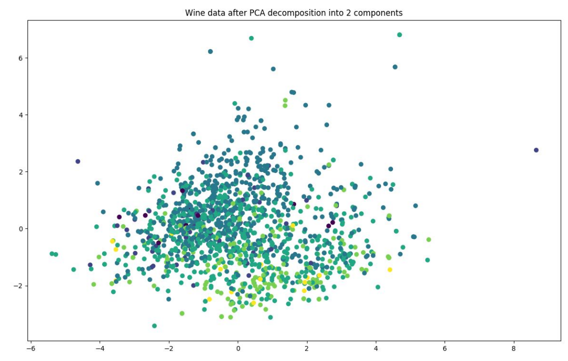
Figure 5. Clusters produced by K-Means (k=6) on the wine data. Centers are shown as black circles.

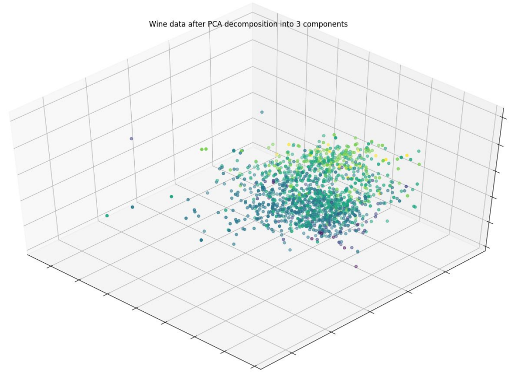
The clusters make intuitive sense, if we imagine a notion of distance or similarity existing between the examples based on their features in this dataset. Unfortunately, looking back at Figure 2, these results do not line up well with the ground truth classes. This shows that the features in this data set do not classify the data well when used as a measure of distance or similarity between examples, which is how K-Means determines what cluster an example belongs to. Expectation Maximization performed worse than K-Means for this data. This tells me that the data is not easily representable with respect to its class as a sample from one of k Gaussians, since one of the principle assumptions that the Expectation

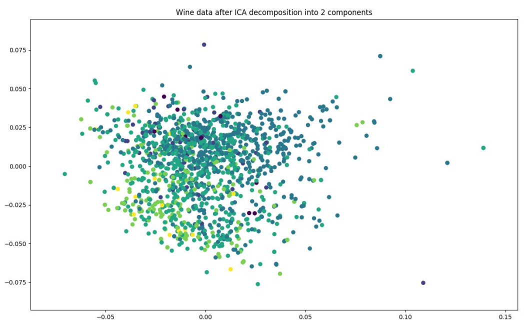
Maximization algorithm makes about the data is that each example is generated from one of k Gaussian distributions.
We see similar performance results with the car dataset, with the exception of the Expectation Maximization algorithm, which dramatically outperformed K-Means in homogeneity and slightly outperformed K-Means in completeness. This tells me that the data from the car dataset is better represented as samples from a Gaussian than using the features as a measure of distance. Given the limited feature space described above, resulting in a sort of “stacking” of data points across the examples, this makes sense to me. Some examples would have a distance of 0 based on the features, but have different labels, hurting the performance of K-Means.

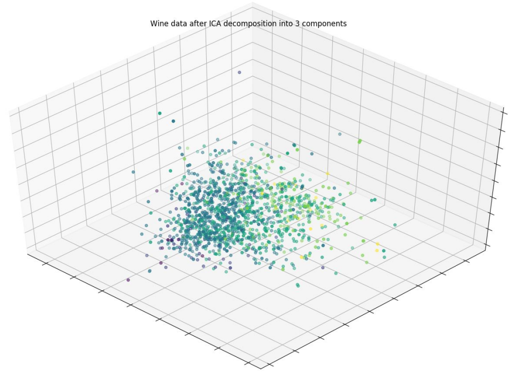
<u>Experiment Two - Dimensionality Reduction</u>

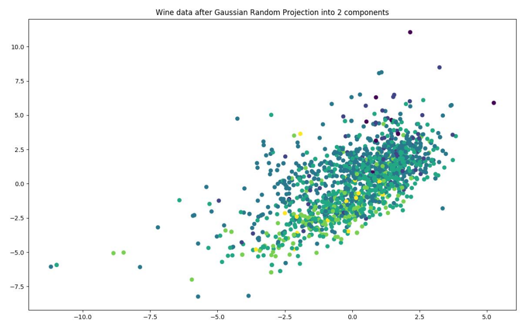
The next experiment I performed was to apply dimensionality reduction algorithms to both of the datasets. The algorithms I used were PCA, ICA, Randomized Projections, and Select K Best Features. Each algorithm takes a parameter which represents the number of components or features to transform the data into. For the purposes of this exercise, I selected values of two and three to use for each algorithm, simply to make it easy to plot without losing any data. I then plotted the transformed data to see how the results compared to the original data as shown in Figures 1 and 2. The results are below.

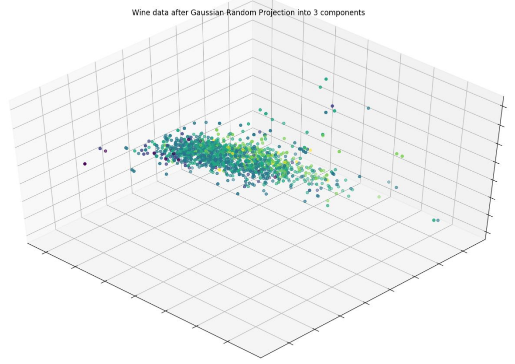












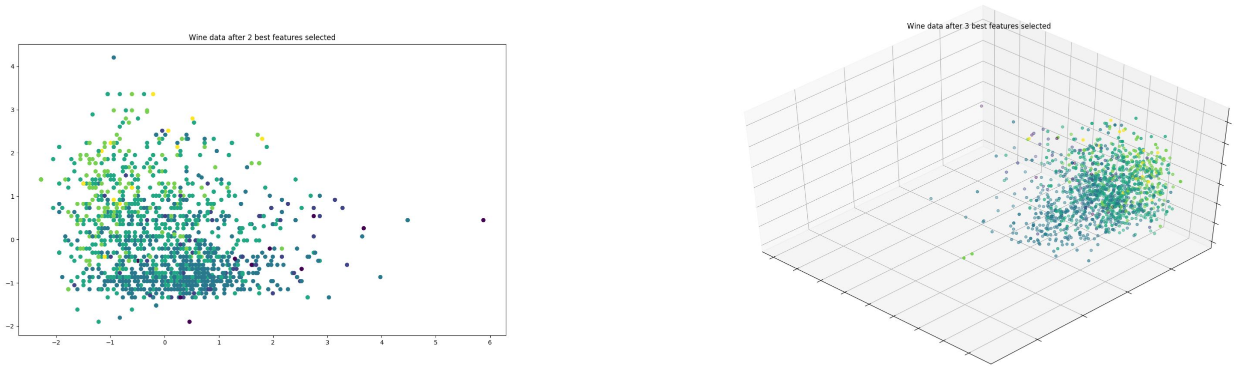
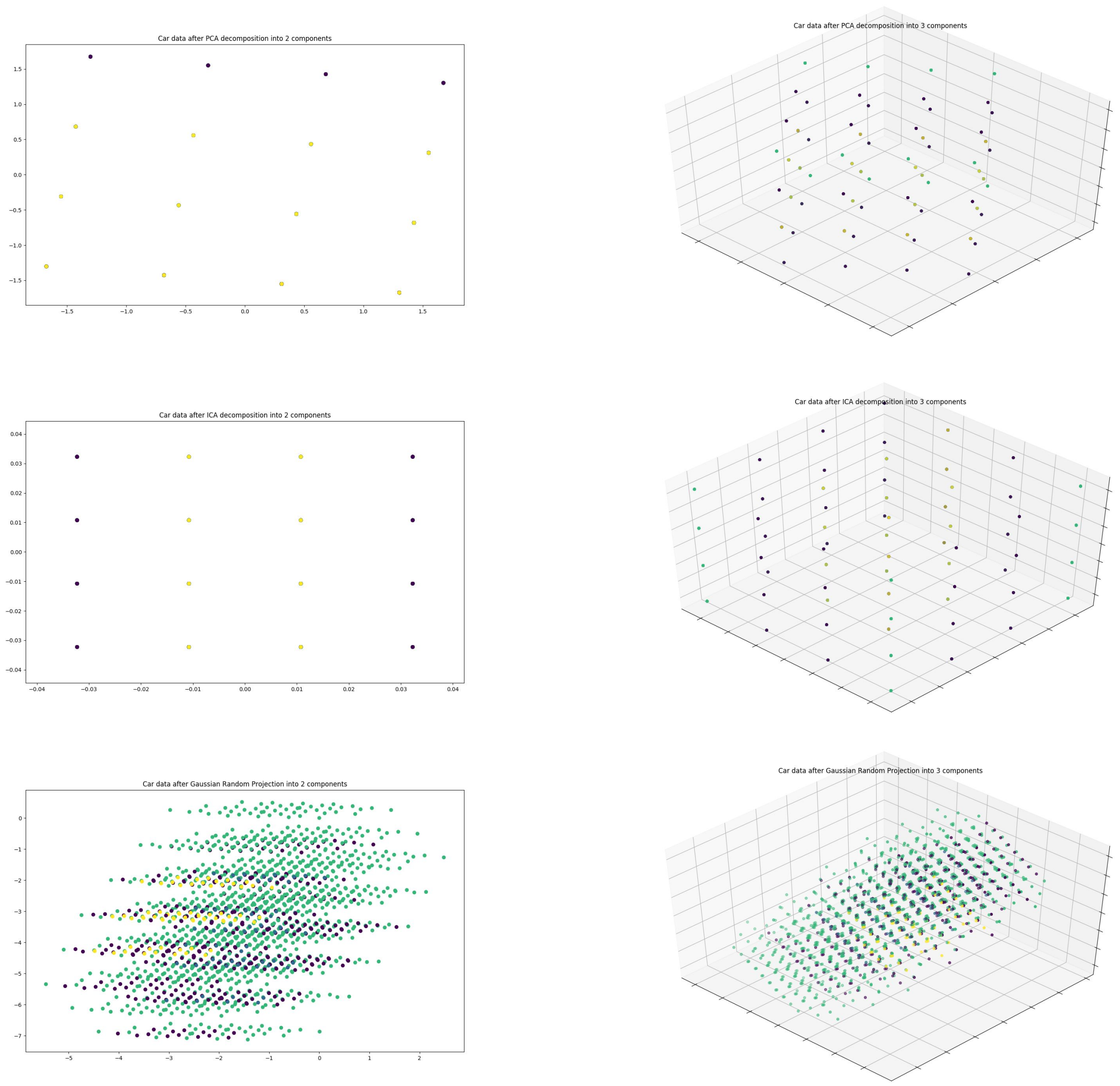


Figure 6. Dimensionality Reduction results on the wine dataset. The color of the points correspond to their class labels. The colors correspond to the same labels as in Figure 2.



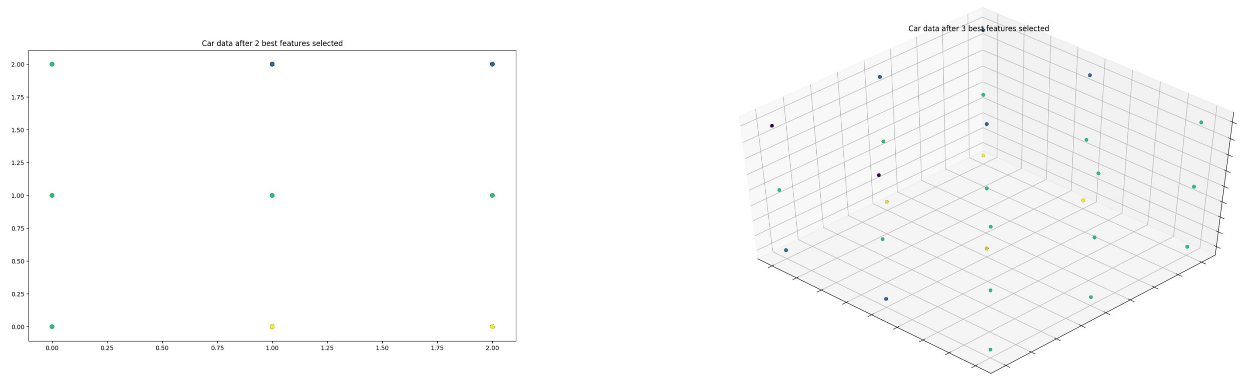


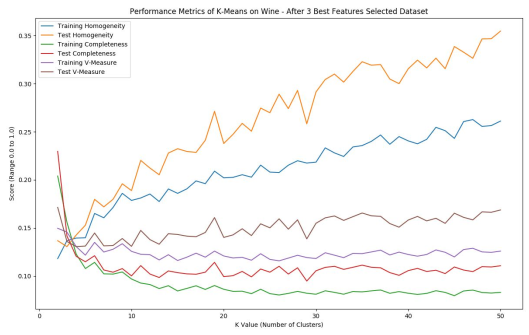
Figure 7. Dimensionality Reduction results on the car dataset. The color of the points correspond to their class labels. The colors correspond to the same labels as in Figure 1.

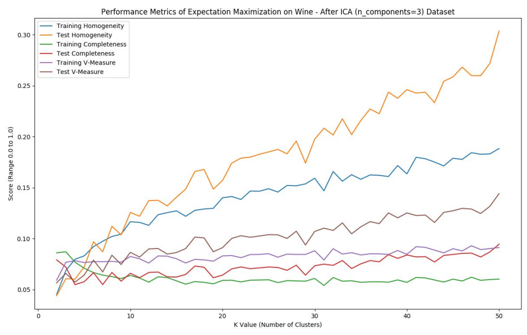
I’ll start by discussing the wine results. The data projected to 2 dimensions by PCA looks similar to the original 2D data shown in Figure 2. The 3D PCA projected data seems quite a bit flatter than the original. In both cases, however, the data points are still fairly mixed together. PCA seeks to select components which maximize variance, which would effectively spread the data out, but this hasn’t occurred for these numbers of components. Presumably, even the components with maximal variance did not have a very high variance for this data. There is a similar result for the ICA projections. ICA seeks to find components that are maximally independent. While the issue with ICA could be that I used too few number of components for this dataset, I think there is a larger problem that this particular dataset does not mesh well with ICA’s underlying assumption. ICA assumes that the data comes from a series of independent sources. The data in our case are chemical properties of wine. Without having an abundance of domain knowledge, I am under the assumption that there are at least some chemical properties that share some relationship with each other. This fact would make it difficult for ICA to effectively reduce the dimensionality of this dataset, especially in a way that would facilitate accurate classification later. The randomized projection results are more interesting. I ran the algorithm several times, and sometimes would get a result similar to the one in Figure 6, with a longer, almost linear resembling distribution of data, and sometimes would get something similar to the original distribution of the data. There was overall a decent amount of variance between these projections. I attribute this to the relatively large difference in starting and ending dimensionalities (11 to 2 and 3). Since the Randomized Projection algorithm simply chooses random directions, there will be more ways to choose these directions if there is a large difference in the start and ending dimensionalities. Finally, Select K Best Features generates a projection somewhat similar to the original data. This algorithm is more of a feature selection algorithm however, and I did not expect the new projection to differ in a meaningful way from the original data. This algorithm simply uses a scoring function and selects the K best features from the feature set based on how well those features predict the correct label.

I thought the PCA and ICA results on the car data were interesting. They both, at least in 2D, seem to have increased the “stacking” effect of the classes I discussed earlier. They both have the same number of samples in the 2D plot, but two fewer classes are represented. The 3D projections are interesting as well. The first thing that I noted was the similarity of the ICA projection to the original. This could indicate that the features in the car dataset behave statistically independent. The PCA projection into 3D is much more compact than the original. Since the data was much more sparsely distributed to begin with than the wine data, I suspect PCA was able to find components with a much larger variance than with the wine data. The random projections had the interesting effect of “unrolling” the stacks of examples from the other projections. The scatter plot looks as though it has much more data, but this is because the examples are not sitting directly on top of one another on the plot. I think this will lead to the randomized projections outperforming the other algorithms in later experiments. Finally, the Select K Best has even fewer apparent data points than the original, which means that even more examples are laying on top of each other in these projections.

<u>Experiment Three - Dimensionality Reduction Followed By Clustering</u>

For this experiment, I ran the four dimensionality reduction algorithms in the same manner as experiment two, with number of components equal to two and three, and then applied the clustering algorithms exactly as in experiment one, generating the same metrics. For the sake of brevity for this written analysis, I will only display results for and discuss the cases of number of components = 3, but I will include the results for number of components = 2 in my repository with my source code. I also only included some of the graphs, as many of the results didn’t differ from the original clustering experiment in a meaningful way. These missing graphs will also be in the source code repository, and I would encourage the interested reader to examine them. Some of the results of this experiment are below.





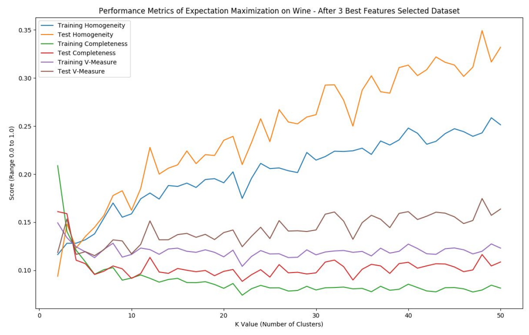
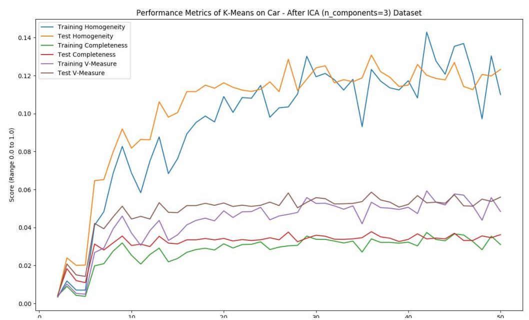
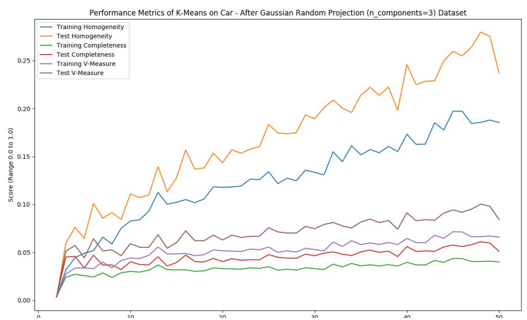
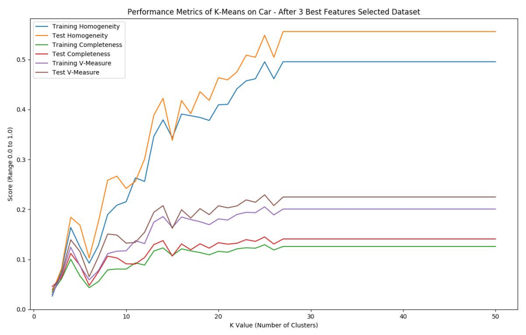
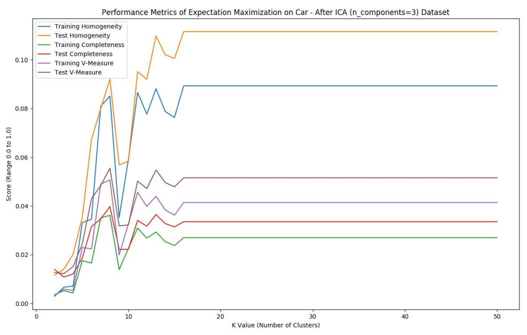


Figure 8. Some interesting results of the clustering algorithms on the wine dataset following dimensionality reduction algorithms.









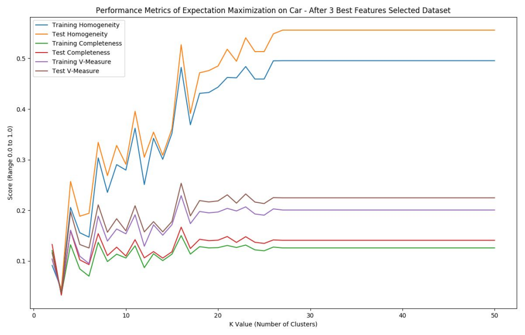


Figure 9. Some interesting results of clustering algorithms on car dataset following dimensionality reduction algorithms.

For the wine data, the PCA reduced data performed almost identically to the tests on the unreduced data, for both K-Means and Expectation Maximization. I was not surprised by this, given how similar the transformed data looked to the original data. PCA didn't do a great job of spreading apart instances of classes in this data, so it makes sense that the performance would be similar. ICA also performed almost identically for K-Means, but interestingly enough it performed slightly better for Expectation Maximization. Randomized Projections also performed similarly to the original experiments. Given that the randomized projection looked quite a bit different than the original projection, while PCA and ICA generated projections that looked similar, I expected Randomized Projections to perform differently than PCA or ICA - either dramatically better or dramatically worse. Looking back at the randomized projection, however, I can see that the original problem with this data remained, despite the different appearance - members of classes were mixed together, without a good way to separate them into very distinct clusters. Select K Best features performed better than the other algorithms, which I expected since it is specifically for identifying the best features for classification. It didn't perform significantly better, however, which I again attribute to the general difficulty of the problem presented by the wine dataset.

The car data produced some interesting results as well. PCA didn't produce much of a different result, despite it producing one of the more dramatically different projections in 3D. Meanwhile, ICA, which produced a result which looked very similar to the original 3D projection, performed significantly worse for both K-means and Expectation Maximization. I was also surprised to see Randomized Projections perform worse than original for K-Means. I suspected that since the randomized projection produced data much more spread out than the original projection, that it would perform much better than the original. This must show that the “stacking” effect I described above did not have as much as a negative effect on the clustering of the data as I previously thought.

Experiment Four - Dimensionality Reduction Followed By Neural Net Classification

I selected the wine dataset for this experiment over the car dataset, as the neural net already performed very well on the car dataset and I was interested to see if the dimensionality reduction could provide any great boost to performance. To perform this experiment, I ran each dimensionality reduction algorithm discussed above on the data to get the transformed data, generated a test and training set, and used a neural net learner to perform both cross validation testing and to generate a final accuracy score on the test set. The number of components used in the dimensionality reduction algorithms ranged from 2 to 10 (one less than the total number of features in the original data). A baseline test was also performed. The results are below.

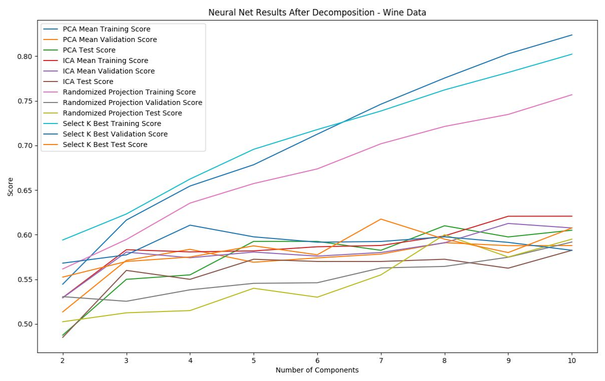


Figure 10. Results of neural net classifier on wine data following dimensionality reduction.

The first thing I noticed was that, in general, the reduced datasets did not perform much better than the baseline, if at all. The baseline test score was 0.5925, while most of the test scores following reduction were in the mid to high 0.5s. I think this could be due to the fact that this dataset was difficult for a classifier to learn to start with, and reducing dimensionality dramatically resulted in a loss of information which negatively affected the neural net learner’s performance. The fact that the neural net generally did better as the number of components increased seems to back this up - as the dimensionality reductions became less dramatic, the learner performed better. PCA and Select K Best were the two best performers in terms of accuracy. Both performed best with the number of components around 7 or 8, with test scores around 0.61 and 0.62. Both of these algorithms consistently outperformed randomized projections and ICA. I expected Select K Best to perform well in comparison to the other dimensionality reduction algorithms, since its specific purpose is selecting features for classification problems. I was a bit surprised with how well PCA performed, especially given that it didn’t perform particularly well in clustering. I was also surprised randomized projections didn’t perform very well - this algorithm was my pick to perform the best besides Select K Best, since it tends to do well with classification problems by retaining much of the original signal while still helping to combat the curse of dimensionality. Cross validation times decreased a bit on the reduced data, which is to be expected - less features means the neural net can train faster. It wasn’t as dramatic as I expected, however. The baseline’s cross validation time was around 26 seconds, and even when reduced to two components the cross validation time never broke below 20 seconds. PCA, Randomized Projections, and Select K Best had very similar cross validation times, usually around 23 or 24 seconds. ICA was consistently faster, usually around 20 or 21 seconds. The cross validation times didn’t vary much as the number of components increased, which was different than I expected. I expected that the cross validation times would increase as the number of components would increase, since dimensionality would be increasing.

Experiment Five- Clustering Followed By Neural Net Classification

The final experiment I performed was to treat the clustering algorithms as dimensionality reduction algorithms and perform the same neural net tests as in experiment four. To generate the transformed data, I fit the clustering algorithms to the original data and produced the predicted clusters as the transformed data. The result is a dataset with a single feature - the predicted cluster. I performed this test for values of k from 2 to 20, inclusive, for both K-Means and Expectation Maximization. The results are below.

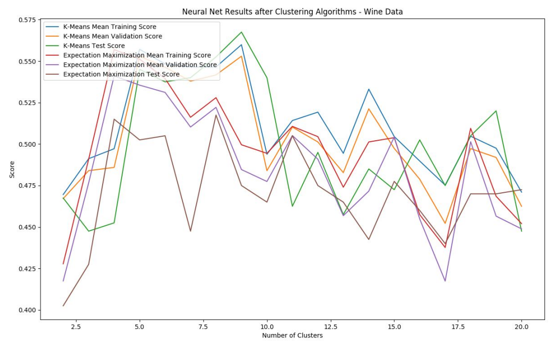


Figure 11. Results of neural net classifier on wine data following clustering used as dimensionality reduction.

I was pleasantly surprised with the performance of the classifier with this data, given that the data has been reduced to one dimension. With the dimensionality reduction to 2 dimensions in experiment four, many of the test scores were below 0.5. The best performing k value in this experiment had a test score of over 0.53. Looking back to Figure 3, we can see that this value of k (around 7) is also where there was a high value of completeness. A clustering with higher completeness would be a better feature for classification, as a perfect completeness score would mean that every instance of a given class is a member of the same cluster. Assuming there is also high homogeneity, this would make for a fantastic feature, as the feature would essentially be the label itself. Like before, with the dimensionality reduction algorithms, none of the scores quite meet the baseline test score. I attribute this to the fact that this dataset is a hard classification problem to begin with, with lots of likely related features. Reducing dimensionality likely loses some information, hurting the performance of the neural network. For low numbers of clusters, the cross validation time was dramatically lower than the baseline, as low as 5 seconds for num_componets=2 for Expectation Maximization. However, as k increased, the cross validation time increased to similar times as the baseline. There was a lot of variance in the cross validation times, however. Excluding k=2, the cross validation time was as high as 24, only two seconds shorter than the baseline, and as low as 10 seconds. The most common value was around 20 seconds, which is an improvement, but like the dimensionality reduction algorithms, not as dramatic as I envisioned.

<u>Conclusion</u>

I was not impressed by the performance of the clustering and dimensionality reduction algorithms on the wine and car datasets. The difficulty of the wine problem seemingly proved to be too great for these algorithms to overcome. The features are likely related in some fashion, making many of the dimensionality reduction algorithms poor choices, and the nature of how the classes were assigned to instances (subjective ratings by humans) makes the classes difficult to learn even with supervised learning algorithms, let alone clustering algorithms. The car data suffered from having a few number of apparently equally important features - throwing away some of them resulted in large losses of information. The distribution of the data looked to be difficult to cluster well, despite the discreteness of all of the features. These experiments emphasize the importance of domain knowledge in picking machine learning techniques to apply. For the most part, for these two datasets, the algorithms discussed were not very effective, despite their success with other types of data.