CS7641 Assignment3 - Unsupervised Learning

Ben Parli

April 3, 2016

Introduction

In this assignemnt we are to implement two clustering algorithms, K-means and Expectation Maximization, as well as four Dimensionality Reduction algorithms, PCA, ICA, Random Projections and SVD. These algorithms will be implemented and run against the Wine Quality and Cars Evaluation datasets, ultimately to understand if a better classification model can be found through feature reduction and feature transformation.

The following sections are organized as follows; the datasets are first described followed by a description of the clustering and dimensionality reduction implementations as well as observations analysis. The second major section will analyze the results of applying the Neural Network implemented in Assignment 1 to the Dimensionality Reduction and Clustering algorithms. Finally, the assignment will conlcude with overall findings and analysis.

Datasets

The classification problems chosen for this course are the Wine Quality dataset and Car Evaluation dataset. Both were downloaded from UC Irvine's public repository and both present examples of interesting and potentially far-reaching applications of machine learning and pattern recognition. Wine quality has been shown time and again to be extremently subjective and even based on visual cues. As a business then, the wine-maker may seek some additional data-driven guarrantee to ensure quality. Further, if patterns can be objectively derived to determine quality based on physicochemical properties in as fickle an industry as wine, there is no reason a similar approach couldn't be applied to industries elsewhere. On the other side of the counter and in a similar way consumers can leverage such Machine Learning output to determine product quality independent of a subjective human "expert." In Assignment 1 this dataset proved very difficult to fit and none of the tuned algorithms were able to perform at better than 25% error rate. All of the features in this dataset are continuous (measurements of physiochemical properties of the wine) while the label for each sample is categorical ranging from 3-8 with higher scores indicating higher quality.

The Car Evaluation dataset is the other dataset explored in Assignment 1. All of the faetures in this dataset are categorical in which each features is essentially ranked from bad to very good. The labels in this dataset are also categorical. This dataset proved to be easily modeled and all of the algorithms of Assignemt 1 performed well. The Neural Network performed with 0% training error and 3% test error. Because of this, the Car Evaluation dataset will not be used in section 2, only the more interesting Wine Quality classification problem.

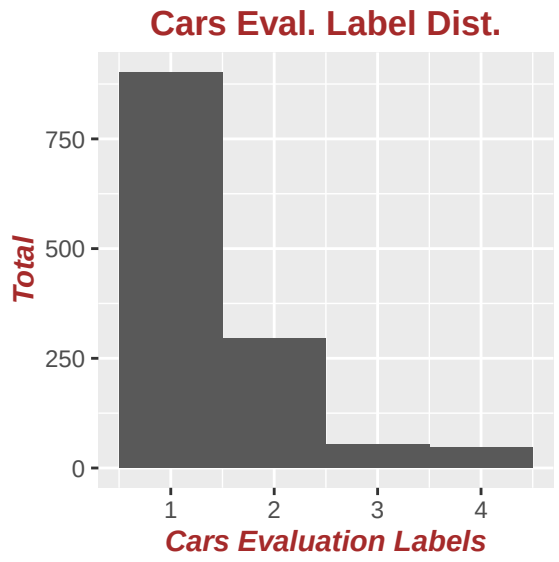
Clustering Algorithms Approach

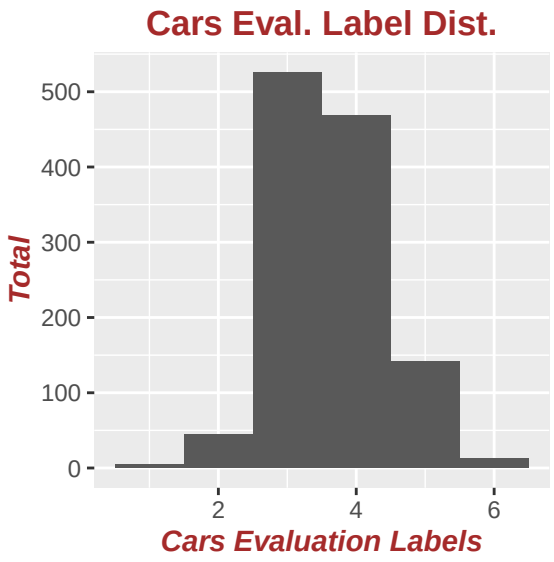
The Wine Quality dataset features are already continuous so little preparation was needed. The Car Quality dataset features were converted from categories on a scale (bad, averag, good, very good, for example) to a scale of numeric scores. In this way both datasets can be treated with the same distance measure, euclidean. In addition, the Wine Quality dataset was scaled to address any features which carry a larger magnitude and thus may unjustly influence the similarity metric. The Car Quality dataset was already sufficiently scaled from the initial data tranformations.

Since both datasets reside in more than two dimensions, it is difficult to visualize the clusters. Instead, the efficacy of the algorithms will be compared and gauged against how well they organize the data. We know from Assignment 1 the Wine dataset classes fell into a Gaussian distribution; that is, most of the wine hovered around average classes with only some being very poor or very good. The Cars dataset on the other hand had a definite left skew; approximately 70% of the cars were labeled unacceptable. We can use this knowledge as a gauge for judging the clustering, but can also use the clustering to ask questions of the original survey and provide feedback. For example, if an optimal cluster of the Cars dataset has 6 clusters, perhaps the original label vector should have also been length 6 instead of 4.

Euclidean distance is used for both datasets. Both datasets contain features which can be considered a range of values where the difference between any two values is essentially the magnitude of disparity. Therefore, Euclidean distance fits the requirement for a similarity measure since feature values which with a large mathematical difference are intended to be represented as more different.

We can see the original distribution of classifcation labels in bot datasets below. In the Cars Evaluation dataset, most of the samples were labeled as unacceptable or just acceptable so the distribution is skewed left. The Wine Quality dataset shows more of a zero-skew normal distribution in that most of the samples fell around an average rating and only some of the wine samples were thought to be bad or very good. These distributions are important to keep in mind as the clustering algorithms are assessed and also as the Dimensionality Reduction algorithms are applied.





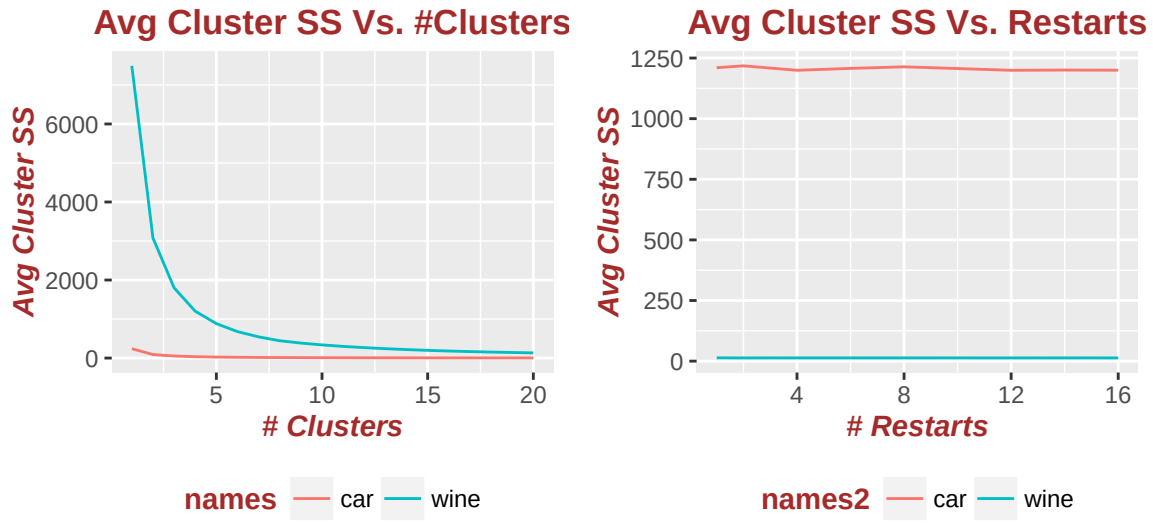
K-Means

The kmeans function in the stats R library was used as the kmeans implementation. Since we use kmeans for analysis of the data, rather than classification, the Sum of Squares is used to gauge and analyse the implementation. We can also compare the distribution of the resulting cluster to the original distribution of the class labels.

Similar to some of the Random Optimization algorithms, restarts are a neccessary component of the kmeans implementation since the original cluster centers are picked at random. It turns out, however, that in the cases of both these datasets, the minimal Sum of Squares was found very quick, in under 10 restarts.

As can be seen below, the mean Sum of Squares per cluster dropped very quickly as the number of clusters was increaed, particularly for the Wine dataset. This is intuitive, since we know the Wine dataset to be more complex and difficult to classify, it is similarly difficult to cluster at small k values. The Average cluster Sum of Squares continues to decrease all the way up past 20 clusters. From the graph, however, we can consider the optimal number to be around 6-10 since thats where the slope really starts to flatten. As k increases and

the slope becomes flatter, the model becomes less and less general. A similar analysis can be done with the car dataset, however, it is easier to see the optimal k value is around 3 or 4.



Expectation Maximization

The em function in the mclust R library was used as the Expectation Maximization implementation. Unlike Kmeans, EM does not need restarts or take it as parameter. Further, this implementation of EM does not take number of clusters as a parameter, instead settling on the optimal number as part of the implementation. For the cars dataset, this ended up being 4; exactly the same as the original label vector and roughly around where the kmeans Sum of Squares slope flattened in the figure above. The Kmeans and EM algorithms seem to be in agreement on the Cars Evaluation dataset then.

For the Wine dataset, the algorithm settled at 6 clusters. This is a bit lower than where the kmeans Sum of Squares slope flattens, but not horribly so. Interestingly, this is also the exact same number as the original label vector. For the more difficult Wine Quality dataset, the kmeans and EM algorithms are generally in agreement, but not as closely as with the Cars Evaluation dataset.

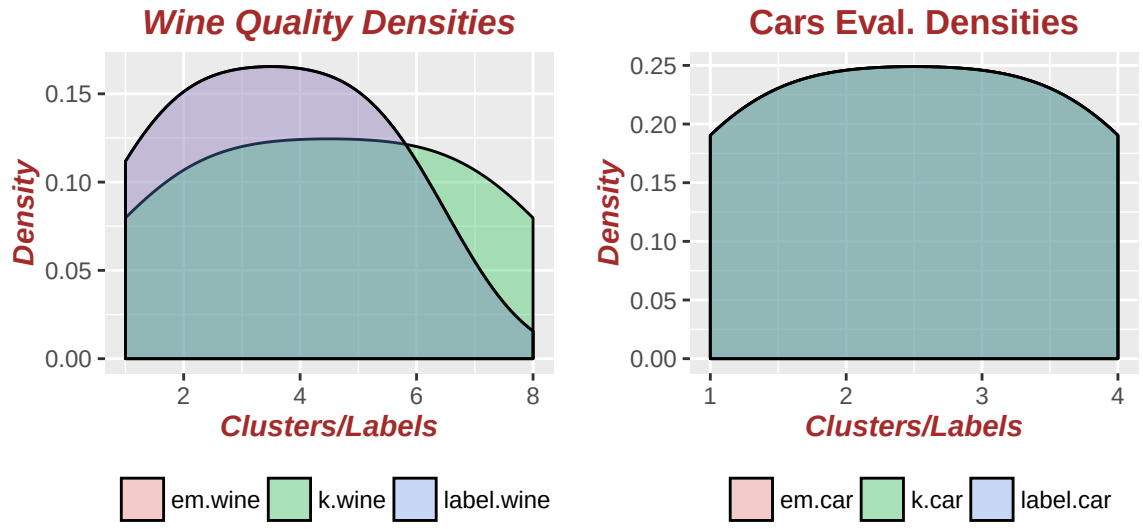
We know that while each sample of the dataset will be assigned to a cluster according to the maximal likelihood, some samples are "corner cases" in which they could very easily be assigned to another cluster. Looking through the EM output shows a few of these cases where the difference in likelihood between two clusters is as low as 1.5%. Since the Wine Quality dataset has proven to be the more difficult to classify we can hypothesize there will be more of these corner cases than in the cars dataset. However, the cars dataset actually had 14.5% of its samples fall into this corner case category (arbritraily defined as having some likelihood between 45% and 55%). Interestingly, the Wine dataset only had 2.5% of its samples meet this same criteria. This outcome may be due to the nature of EM and these datasets though. We have already seen from the first figure above, that the Cars Evaluation dataset is skewed left whereas the Wine Quality dataset is more of a zero-skew normal distribution. Since one of the fundamental assumtions of this em function in the mclust library is a Gaussian distribution, the Wine Quality dataset should lend itself more to the EM algorithm than the Cars dataset.

Clustering Comparison

Another way we can compare the two clustering algorithms is by how they organized the data and also how that organization compares to the original label distribution. The distribution of classification lablels is density plotted alongside the two clustering algorithms for comparison. As we can see below, the clustering

algorithms are in complete agreement on the car dataset distribution. There is nearly complete overlap and the original labeled data distribution is also in agreement. Of course, the algorithms do not consider labels, however, that they organize the data exactly alike indicates harmony.

The same was done for the more difficult dataset, the Wine Quality dataset. In this case the classification label distribution is in agreement with the EM algorithm in terms of clusters. We can see the kmeans density is not as well aligned with either the EM algorithm or the classification labels, but in large part because we identified 8 clusters as the ideal from the Kmeans Sum of Squares graph. Looking at the distributions of the k.wine plot, we can see the edges fall off quickly. Looking at the number of samples in these "edge" clusters also indicates a better number of clusters for kmeans may be 6 instead of 8. In re-running the experiments and re-plotting the densities with 6 labels, 6 kmeans clusters, and 6 EM clusters, we arrive at a (boring) plot in which all desnities are in agreement and overlapping. This isn't plotted here due to its simplicity, but the upshot is the kmeans clustering is in agreement with EM.



Dimensionality Reduction Approach

Four Dimensionality Reduction algorithms are used against the Wine and Car datasets. The princomp function in the stats R library was used as the PCA implementation, the ICA R library and icaimax function as the ICA implementation, a custom R script as the Random Projections implementation, and the SVD R library and svd function as the SVD implementation.

The wine and Cars dataset produced the below principle components with the output indicating standard deviation of that component. This is also indicative of the eigenvectors distribution since squaring each standard deviations equals a value proportional to the eigenvalue. We can see in all the algorithms run against the Wine Quality dataset, the first critical component/projection is clearly the most important. There is a steep decline in importance from that first component, but some of the remaining are needed to explain the data as the first by itself would not be enough to explain the data.

In seeking to reduce the dimensions of the Cars dataset the first three principle compnents supply the most information. There is a dropoff from the third to the fourth so the ideal cutoff would be at 3 principle components for the Cars Evaluation dataset. This is contrary to the Wine PCA output; the first three Cars principle components are close to one another in terms of standard deviation.

```
wine.pca$sdev
```

```
##     Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6
## 0.26724478 0.19549126 0.17567726 0.12305128 0.10489533 0.10130333
##     Comp.7     Comp.8     Comp.9    Comp.10    Comp.11
## 0.08838181 0.08253359 0.06955269 0.05405749 0.03466187
```

```
cars.pca$sdev
```

```
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6
## 1.1355837 1.1233236 1.1124448 0.8205770 0.8164647 0.8063393
```

Similarly, the Wine Quality ICA algorithm shows a very steep dropoff in variance accounted for from the first component. We would need the next 5 components to explain another 50%. The outcome is similar for the Cars Evaluation ICA implementation; the first three components exaplain most of the variance, over 60% in this case. If additional variance is needed an additional component can be added at a time since the remaining 3 all account for ~11% of the variance.

```
wine.ica$vafs
```

```
##  [1] 0.24332681 0.12562731 0.11478051 0.11348173 0.09848192 0.07659890
##  [7] 0.05759945 0.05073811 0.04384504 0.04111728 0.03440294
```

```
cars.ica$vafs
```

```
## [1] 0.2213439 0.2188620 0.2152150 0.1153827 0.1152526 0.1139438
```

The SVD diagonal matrix values also show similar attributes in that the very most important is the first value, however, we can see the importance of the next three values as well. From there the contribution flattens. The Cars SVD output deviates a bit from PCA and ICA in that the dropoff in importance occurs after the first column.

```
wine.svd$d
```

```
##  [1] 33.714051  8.798921  6.755894  5.976238  3.919333  3.522080  3.214655
##  [8]  3.060254  2.409020  1.895375  1.227806
```

```
cars.svd$d
```

```
## [1] 202.78293  40.87530  40.26142  34.31732  29.46793  29.07359
```

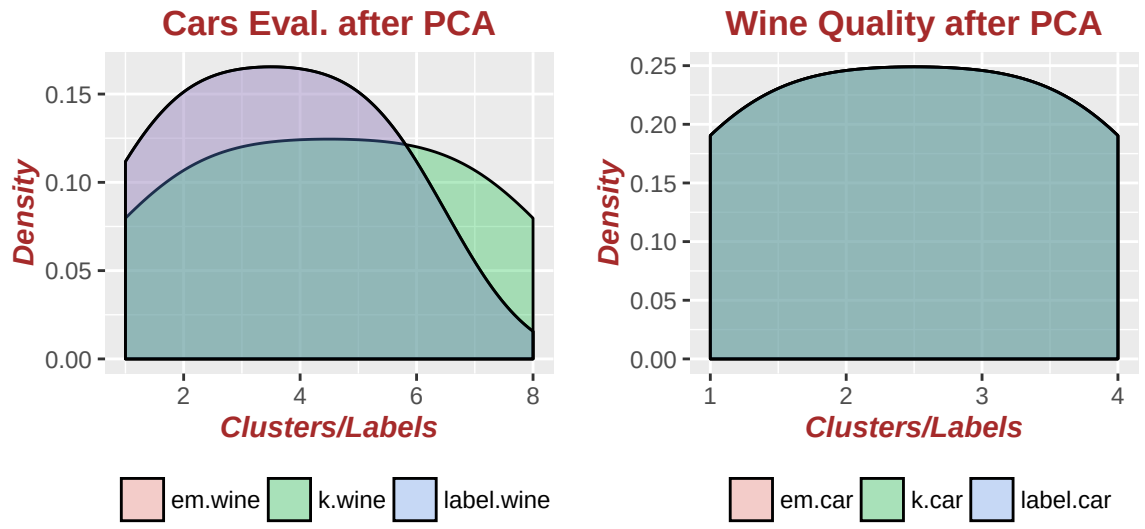
The Random Projections algorithm is a different approach from the above 3 since the algorithm arrives at the number of features taken as input. The algorithm was iterated over 100 times while seeking to reduce to 6 dimensions in the Wine Quality dataset and 3 in the Cars Evaluation dataset. The Wine Quality dataset is used to gauge the effectiveness of this algorithm more below.

The question, of course, with all these Dimensionality Reduction algorithms is does the variance explained by the reduced number of transformed dimensions explain enough of the data. From the outputs above a logical cutoff can be identified to explain most of the variance. However, for a dataset such as Wine Quality, which has proven difficult to model, is that enough? Will more of the compnents be neccessary even though they only contribute very little to the overall model? Only though trial and error can this be answered.

Clustering after Dimensionality Reduction

The graph of Kmeans sum of squares within each cluster for the clustering performed on the reduced dimesnions is very close to the original. Additionally, the EM algorithm arrived at the same number of clusters on the reduced dimensions compared to the original. The desnity plots are also extremely similar. This all indicates these dimensionality reductions are appropriate and effective for the cars Evaluation and Wine Quality datasets. That is, we can perform the PCA dimesionality reduction, continue with the clustering algorithms and still arrive at a similar quality of outcome. In both cases the first three principle Components explained enough of the variance for the end result to be very close.

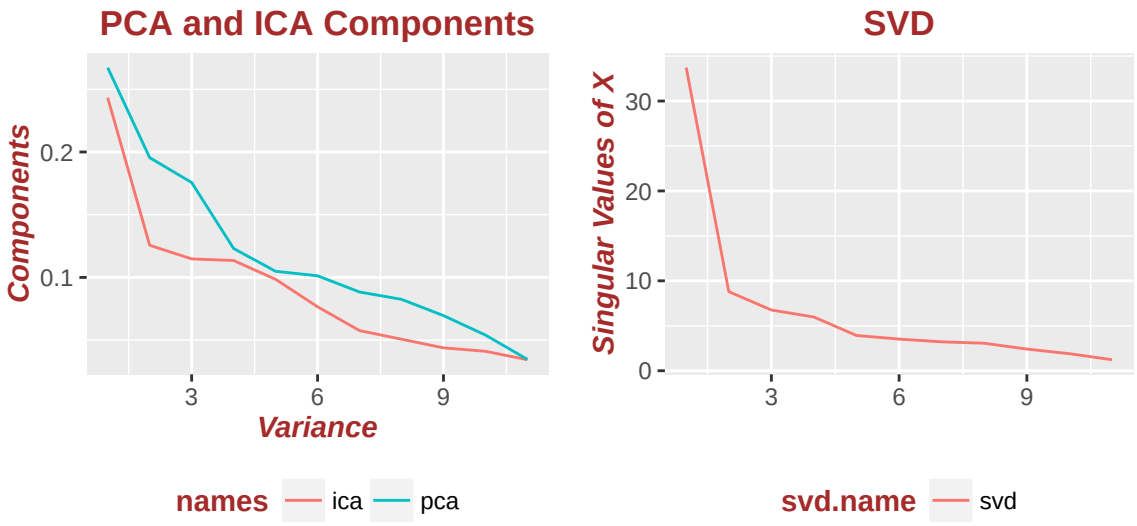
The figures below show overlaid density plots of the original class labels and the two clustering algorithms after PCA Dimensionality Reduction has been performed. Again, the Cars Evaluation dataset is nearly perfectly overlapped, indicating we should be able to map a label to either an EM or Kmeans cluster. Similarly, for the Wine Evaluation, the EM algorithm is nearly perfectly overlapped.



Revisiting Neural Networks with Dimensionality Reduction

In order to gauge the effectiveness of each Dimensionality Reduction algortihm the number of compnents used will be the same for each. The original tuned Wine Quality dataset Neural Network identified in assignment 1 contained 12 hidden nodes with no set bundaries on the weights so that will again be the model here. Some changes from this setup were tested in each case but in most cases didn't show much difference in performance or the performance slightly decreased.

The PCA implementation is described in the section above, but this time using the first 6 principle components instead of 3. This will only serve to increase the variance contained in the final model. This implementation finds independent signals by maximising entropy and conveniently also outputs the variance accounted for by each component.



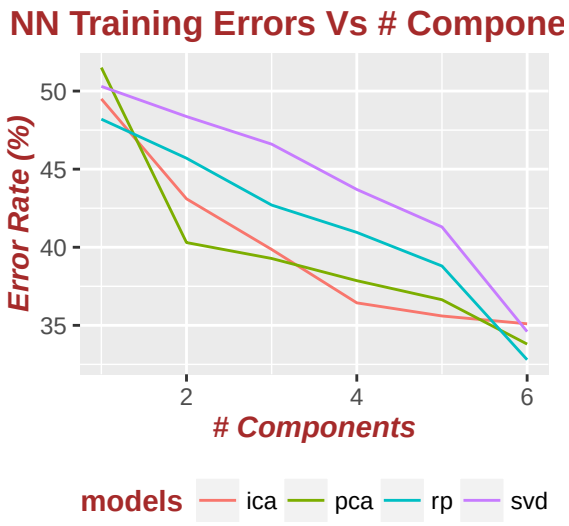
The Random Projections were implemnted by hand. Being a randomized algorithm, a series of 100 iterations were performed and interestingly the variance among the outcomes was relatively small (0.0003785937) with the error rate settling at 36.1%. The algorithm is still seeking 6 components like the others; we can take this as further evidence that a lot of the variance and information can be captured in 6 components (transformed and reduced from 11). The side by side comparisons of the four Dimensionality Reduction algorithms is in the table below.

###Neural Net Performance Table

Model	Iterations	Train Error
1st NN (12 Nodes, 11 feats)	~800	28.7%
PCA NN (12 Nodes, 6 feats)	~650	33.8%
ICA NN (12 Nodes, 6 feats)	~350	35.1%
RP NN (12 Nodes, 6 feats)	~550	32.8%
SVD NN (12 Nodes, 6 feats)	~700	34.6%
ICA NN (12 Nodes, 11 feats)	~440	25.1%

The final entry in the table is a very interesting case. The ICA algorithm stands out from the others in that it seeks to transform the original components into a statistically different dataset rather than purely reduce dimensions. But what if this transformation to a full set of 11 mutually independent features was modeled? Surprisingly, this transformed dataset actually yielded an improvement from the original when the Neural Network was applied to it. This experiment was run 30 times and the ICA transformed set performed better by at least 2% every time. Since the ICA algorithm results in independent components with no shared mutual information, we can see that by transforming and separating these Wine Quality features into new independent features, we are able to gain some additional performance.

The variance described by the first component was discussed above and hypothesized to not provide enough information to adequately model the Wine Quality dataset. We can see that to be the case below as the error rate continues to drop with each additional component. This is further evidence to suggest a reduced number of dimensions is adequate with this dataset. In the case of the Wine Quality dataset we can take this to be an indication the difficulty of the classification problem is not a result of too many dimensions.



Revisiting Neural Networks with Clustering

Next we use the EM and Kmeans clustering algorithms to act in a dimensionality reduction reduction capacity. The approach in both cases is to cluster the Wine Quality data based on whats already been learned so far. That resulting cluster organization becomes our new, single dimension such that each sample with its original n features is now described by one feature. The idea is the single feature should map with some consistency to the labeled classification on the original dataset. The nnet R library Neural Network implementation is used to find this model.

We know from the initial Kmeans implementation that the Sum of Squares starts to level out around 6-8 clusters. The resulting Neural Network achieved a 44.7% error rate. Tweaking the number of clusters in the kmeans implementation yielded nothing above this metric, nor did any attempts to tune the Neural Network. Attempts to simplify the network (as few as 1 hidden node) and attempts to increase complexity (as many as 20 hidden nodes) did not deviate much from this error metric.

The EM algorithm settles on 6 clusters, which could conveniently map cleanly to the number of labels in the original dataset. However, the resulting Neural Network achieved only a 42.9% error rate. Attempting the same tweaks to the Neural Network as for the kmeans experiment did not increase performance.

There a couple key takeaways from this experiment: * Despite a less than ideal model performance, in both cases the classification was still impressive given the number of dimensions were reduced to only 1.
Both clustering algorithms arrived at a similar conclusion in terms of organization of datapoints and number of clusters. The resulting mapping to the original classification labels could inidicate some noise in the data. We know tastes in wine to be fairly subjective. These results would seem to indicate that some samples which should be clustered together under the same label were actually assigned different classes. That is, from these experiments we can hypothesize some of the noise in the data is due to the subjectivity of the dataset topic (wine preference) itself.

Final Takeaways

There are some key observations we can takeaway from these experiements.
Both clustering algoirthms arrived at similar distributions of labels and agreed with the original dataset label distribution. One way these algorithms could be used is to identify the right number of labels or act

as a feedback to the original data collector. * While using Clustering on these two datasets provided good analysis, it did not prove useful in building a usable classifier. That is not wholly unexpected though, since these are Unsupervised Learning methods. * Its not addressed in this assignment, but interesting to note nonetheless; the Neural Network time complexity decreased with the reduced dimensions. As the number of features fed into the Neural Net algorithm fell, so too did the time to converge, although, as we saw above, the resulting classification performance suffered.

References and Citations

R Package nnet, Feed-Forward Neural Networks and Multinomial Log-Linear Models, https://cran.r-project.org/web/packages/nnet/nnet.pdf

R Pakcage ICA, https://cran.r-project.org/web/packages/ica/ica.pdf

R Package SVD, https://stat.ethz.ch/R-manual/R-devel/library/base/html/svd.html

R Package stats, https://stat.ethz.ch/R-manual/R-patched/library/stats/html/princomp.html

R Package Mclust, https://cran.r-project.org/web/packages/mclust/mclust.pdf

R Package Kmeans, https://stat.ethz.ch/R-manual/R-devel/library/stats/html/kmeans.html

UCI Machine Learning Repository, Wine Dataset https://archive.ics.uci.edu/ml/datasets/Wine. Irvine, CA: University of California, School of Information and Computer Science.

UCI Machine Learning Repository, Car Evaluation Dataset https://archive.ics.uci.edu/ml/datasets/Car+Evaluation. Irvine, CA: University of California, School of Information and Computer Science.

Some code borrowed and tweaked from https://github.com/chappers/CS7641-Machine-Learning/tree/master/Unsupervised%20Learning