Sagar Laud CS 4641
Assignment 3

Unsupervised Learning and Dimensionality Reduction

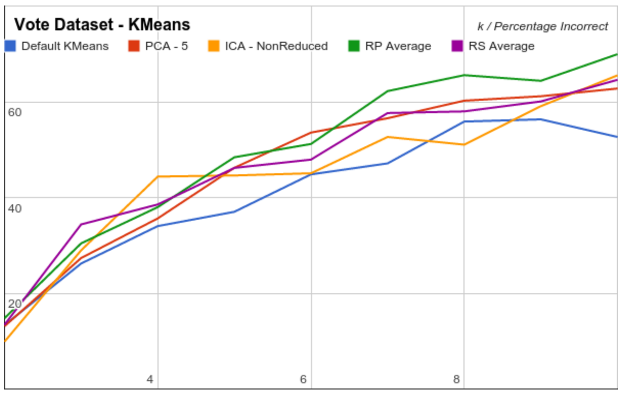
Datasets:

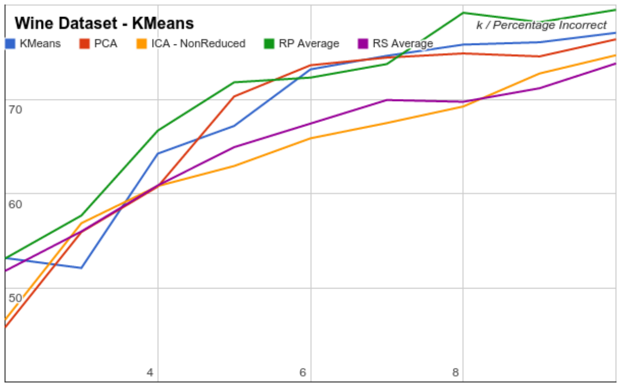
The datasets used for this assignment are the same ones that were used for assignments 1 and 2. The first one focuses on how members of congress voted on certain issues and how those votes align with their political party. The second dataset focuses on the quality of red wine given a slew of chemical properties. In terms of the values of k which were chosen, the values ranged from 2-10. The reasoning for this was because the vote dataset has 2 possible values for it's class attribute but may perform better with larger amounts of k while the wine dataset has 6 possible values for it's class attribute and may perform better in either a lower value for k or a higher value for k as well.

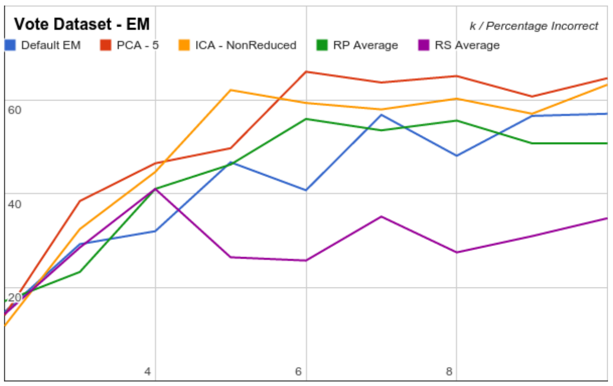
Definition of Correctness:

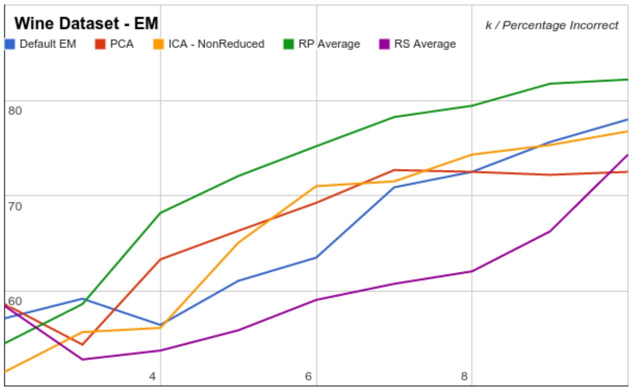
For the sake of defining this though it may seem intuitive, an instance is considered to be clustered properly when the cluster that it is in aligns with it's class attribute. In addition, when an improvement or increase in inaccuracy is stated in any of the analysis in terms of error rate, it is the difference between the best results between the algorithms. In most cases, the best result was produced when k was 2. If this is not the case, then a k will be specified. If no k is specified, then assume that k is 2.

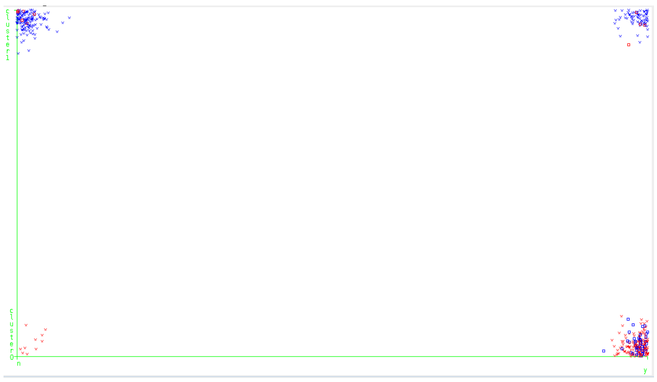
Results:



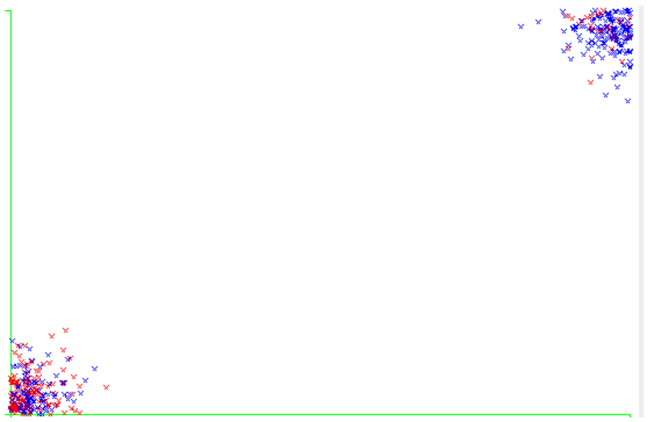








Religious groups in school vs cluster (blue are democrats while red are republicans) in the vote dataset. Most clusters were like this regardless of issue so I chose one. Note how democrats are misclassified far more than republicans. I believe that this is because the dataset contains more democrats so when something is on the fringe between the two classes, it could be classified as a republican. PCA and ICA should be able to work well with this.



This clustering shows how democrats and republicans voted on just one of many issues (this one is for handicapped infants). As shown by this cluster, there is a huge overlap between the two class values for this particular attribute. Things like this will make the results of PCA and ICA interesting because they try to eliminate useless data.

Unmodified Clusters:

Voting Dataset:

The results that we see for both KMeans and EM for the vote dataset are what was expected. Here, the default algorithms worked reasonably well but increased in error as the number of clusters increased. For both algorithms and for the dataset as a whole, the data seems to be best divided into two sets which makes a considerable amount of sense considering the class attribute has two possible values: Republican and Democrat. In addition, because one class does not have an incredibly significant majority in terms of instances, classifying into two classes still worked best without any dimensionality reduction whatsoever. The time taken to run the actual algorithm was virtually zero with only 7 iterations being required to find a solution. EM did not fare any better than classic KMeans even when allowed to pick its own value for k. The algorithm still fared best when k was set to 2. Since both KMeans and EM chose a value of 2 for k, it is a fair bet that this ended up being the best of all values of k for this particular dataset. In terms of data, KMeans clustered best with a k of 2 with 13.3333% error with respect to clusters matching the class attribute. EM resulted in a minimum of 14.7126% error, again with an optimal k of 2.

Wine Dataset:

In terms of the results for the wine dataset, they too make sense though they are not quite what was expected. One would expect initially that the clustering algorithm would work best when k is set to the number of possible values for the class attribute, 6 in this case. However with a value of 6 for k, both algorithms performed abysmally. Of all the possible values for k, 2 seems to be the best for both KMeans and EM. In terms of KMeans, this actually makes a great deal of sense once we look at the distribution of instances within the classes themselves. The instances are highly concentrated within 2 of the 6 classes. As a result, a value of 2, 3, or 4 for k would seem to work best simply because the instances could be clustered such that it either lies within one of these two classes, or does not and lies in one of a few smaller clusters. It is just because of the sheer number of instances in those two classes that I think the value of k worked out to be what it was. KMeans produced an error rate of 52.0951% with a k of 3 while EM produced an error rate of 56.4103% with a k of 4.

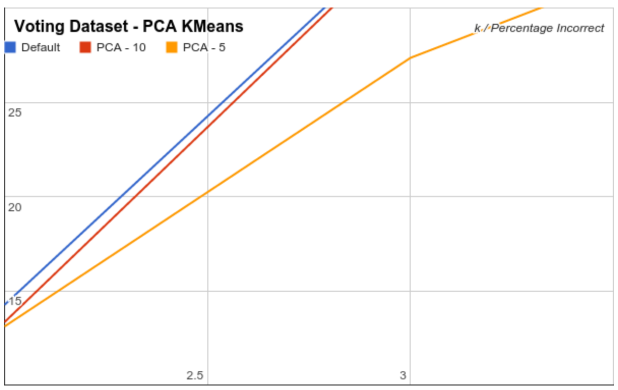
Analysis:

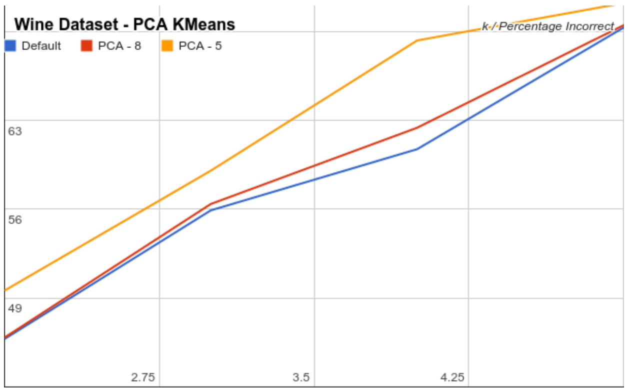
In the vote dataset, pure EM resulted in worse results for every value of k. In the wine dataset however, this was not the case as EM outperformed KMeans when the value of k was greater than 3. The reasoning that I see for this is that EM allows for a little gray area as in points are not definitively assigned to a cluster. So, as the value of k increases, assigning these points which are in the gray area happens based on probabilities and may not always result in the best assignment. In fact, because EM assumes that every single point could belong to every single cluster, as the number of clusters increases, this could result in an incorrect assignment simply because EM opens the realm of possibilities which may not always be the best idea. We may end up allowing an assignment which would never have happened via KMeans and while that may be a good thing sometimes, it proved to be a bad thing in the vote dataset.

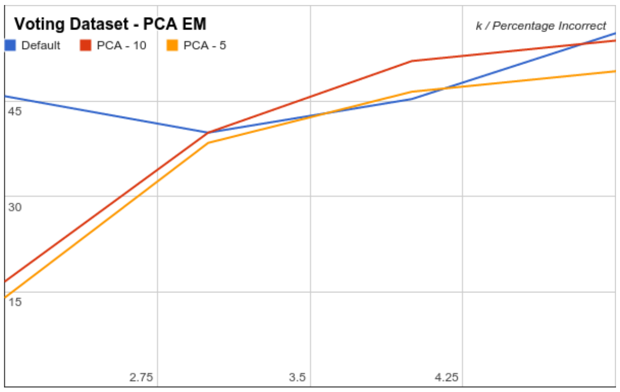
For the vote dataset where the two classes are distinct, allowing a probability for a democrat to be a republican and vice versa could result in an incorrect assignment. The same concept however does not apply to the wine dataset which is why I feel that EM outperformed KMeans given a large enough value for k. The classes in the wine dataset are not quite as distinct. There is not a significant difference between the class attributes as they numerically describe the quality of the wine. Because the classes are not as distinct, the gray area that EM allows for and the fact that it clusters based on probabilities such that every point can belong to any class in theory seems to have removed some of the incorrect classifications by using soft clustering instead of hard clustering. With such a large number of instances existing in two classes in the wine dataset, allowing for this gray area could allow instances that would have been misclassified via KMeans to be assigned to clusters simply because that assignment was deemed to be possible via a probability. While EM is a good idea theoretically, there are certainly situations such as with the vote dataset where a hard clustering proves to be better than a soft clustering simply because in some cases, a naive approach like KMeans removes the extra complexity that leads to incorrect assignments in EM. The extra complexity however can result in better clusters when the classes are not as well defined such as in the wine dataset.

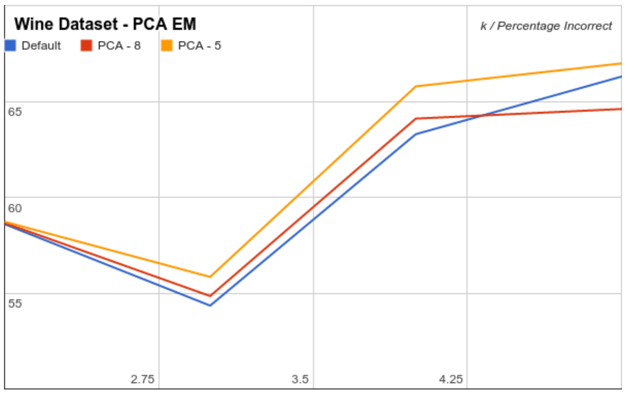
PCA:

Multiple experiments were run with both datasets using PCA. The results are depicted below. Splitting the data into combinations of attributes, thereby skewing the way we looked at information allowed us to remove things based on variance and reduce our dimesionality.









The following values are the eigenvalues of the vote and wine dataset respectively mapped to the combination of attributes that generated that eigenvalue. For PCA-8 and PCA-5, we simply took the first 8 and 5 elements with the highest variance respectively.

Vote Dataset Results:

1	2	3	4	5	6	7	8	9	10	11	12	13
7.27	1.35	1.11	0.91	0.79	0.76	0.61	0.54	0.50	0.44	0.39	0.33	0.30

Wine Dataset Results:

1	2	3	4	5	6	7	8	9
3.1	1.9	1.6	1.2	1.0	0.7	0.6	0.4	0.3

Variance of Eigenvalues:

Dataset	Vote	Wine
PCA	3.45	0.790
PCA-8	4.29	0.787
PCA-5	7.80	0.701

Vote Dataset Analysis:

When analyzing these results, we notice a few interesting things. First of all, the algorithm works best on the vote dataset when the dimensionality is reduced the most. Meaning, the less features present in the data, the better the clustering algorithm is able to group instances together. This is significant as it means that PCA actually worked quite well on this dataset. As combinations of attributes with lower eigenvalues were removed, the correctness of the clustering algorithm increased. It is also interesting that this correlates with an increase in the variance of the corresponding eigenvalues. As the variance in the eigenvalues increases, so too did the best result that was produced for the dataset (both in KMeans and EM). Reducing the dimensions seems to cause an increase in accuracy because whatever information we are losing seems to just be noise. As in, the information that we are losing does not actually help our algorithms and in fact causes them to cluster incorrectly and group them with the wrong class attribute. We can also see that PCA found one combination of attributes which provided a lot of information resulting in an eigenvalue of 7.27. The rest of the attribute combinations still provided some information with eigenvalues of around 1 but none was more informative than this large eigenvalue so it makes sense that as we eliminate combinations of attributes which provide next to no actual information, our results get better and therefore correlates to an increase in the variance of the eigenvalues. Both algorithms improved when run with PCA vs just the default settings and chose a value of 2 for k as the best value, but not necessarily at first. In fact, running PCA and allowing for enough attributes to cover the desired variance actually resulted in worse results, raising the inaccuracy from roughly 13% to over 14%. However, as more and more noise was removed, PCA-5 ended up producing the best overall results with an inaccuracy of 13.1034% which is marginally better than the inaccuracy of 13.333% that was produced from traditional KMeans. In addition, EM also performed best with PCA-5 with an error of 14.023% which is less than the initial error of 14.7126%. That being said, without any reduction in the number of attributes, default PCA when used with EM produced some incredibly bad results. Specifically, the inaccuracy for the best value jumped up to 45%! I think this occurred because of how the attributes were reconfigured. PCA seems to have blurred the lines between the classes slightly which resulted in a large number of incorrect classifications by the algorithm.

Wine Dataset Analysis:

Interestingly enough, we see the exact opposite occur with the Wine dataset. As the variance within the set of eigenvalues drops, so too does the accuracy when we lower the number of dimensions. We see a steady increase in the amount of error when the number of dimensions are reduced in fact. This seems to be because we are losing information as we reduce dimensions and what we are losing is actually important to the clustering algorithm. In this case default PCA ended up performing the best out of all three instances of PCA which were run and both managed to produce better results than the default KMeans and EM with an error rate of 45% and 56% respectively instead of the initial error rates of 53% and 57%. This value for KMeans was actually the best value found overall for the algorithm. I would attribute the huge increase in accuracy in KMeans to be due to the fact that the information was reconfigured but was still grouped using a hard clustering. In doing this, the classes seem to have been spread out a little more resulting in hard clustering working the best. When removing information, the clusters seem to have gotten closer in space resulting in a worse error rate. That being said however, this spacing out of classes still did help EM with fewer points which fell in the gray area still resulting in a better classification rate than traditional EM without PCA.

ICA:

The following table shows the absolute value of the kurtosis for each of the sources rounded to the nearest tenth. All red values were negative originally. Also, all values have been scaled. In a normal kurtosis analysis, 3 is considered to be gaussian. The values have all been scaled to zero such that zero is considered to be gaussian. Two things were tested for this portion of the assignment. ICA was first run on the data and then put straight into the clustering algorithm (without removing any sources). Then, any sources that seemed reasonably gaussian were removed in an attempt to lower the dimensionality more and improve the results. The runs with ICA (without removing any sources) produced the best results for both datasets. The reasoning and data to support this are below.

Vote Dataset:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0.4	1.4	0.2	0.7	0.8	0.2	1.0	0.7	0.1	0.2	0.8	3.9	0.3	1.7	0.8	0.5

Just for the sake of attempting to remove something to reduce the dimensionality, Sources 1, 3, 6, 9, 10, 13, and 16 were removed as they was the closest to being gaussian. This however did not lead to an improvement. The reasoning for this seems to be that the dataset itself is not at all gaussian. When ICA is run and the sources are pulled from the data, the distribution within the sources themselves are not gaussian either. As they are not entirely gaussian, removing any source removes a significant amount of information that the clustering algorithm used. Therefore, for this dataset it is best to leave all of the sources as we cannot reduce the dimensionality using ICA without increasing the inaccuracy of our clustering algorithm.

However, ICA without removing any of the sources produced the best results for both the KMeans and EM algorithms reducing the initial error rates from roughly 13% and 14%

respectively to roughly 10% and 12% respectively. The reasoning that I have for this is that nothing about the dataset at all is gaussian but things are naturally independent. The distribution within attributes or the class attribute isn't gaussian. However, once we pulled the information out into mutually exclusive sources, the information that was provided to the clustering algorithm seemed to be more spread out, which makes sense given that the sources are now mutually exclusive. Because of this increase in space between clusters, both hard clustering and soft clustering worked well as the proper class was typically closer and the amount of gray area was reduced.

Wine Dataset:

1	2	3	4	5	6	7	8	9	10	11
2.0	0.2	1.9	7.5	3.9	6.7	1.3	2.1	0.6	1.3	1.5

The same concept was displayed here. The wine dataset is not gaussian at all. In fact, the majority of the instances are from one of two classes (with a total of 5 classes in the dataset). When ICA was run, nothing was really very gaussian so removing sources 2 and 9 did not produce better results.

However, running default ICA through the clustering algorithm yielded the best results for EM and the second best results for KMeans. For EM, there was approximately a 51% error rate with respect to clusters aligning with their class attribute. This is significantly less than the default EM error rate of 57%. The spacing out which was discussed in previous sections with ICA pulling our mutually exclusive sources seems to have done the trick here. The spacing out of clusters resulting in less gray area combined with EM seems to be a great combination and has clearly worked well. For the KMeans algorithm however, ICA did not beat out the default PCA algorithm. I believe that this is because in this dataset with how close the points are with respect to each other and the fact that the instances are dominated by one of two class variables, removing "noise" is more important than spacing the clusters out a little more. It seems that the dataset had enough noise that when this was brought out to independent sources, this noise was still present and caused slightly worse results.

RP:

Naturally this algorithm works by forcibly reducing the number of dimensions, usually with information loss.

Vote Dataset:

Randomized Projections did not work particularly well for the vote dataset. However, it did not fail as miserably as I expected it to. It did not outperform any of the other algorithms until k reached a value that seemed silly to use but was just tested to observe the results. Randomly projecting the dataset into a lower space still seems to have worked reasonably well with this dataset simply because of the nature of the attributes. All of the attributes provide a reasonable amount of information as most politicians will still vote along party lines. So, projecting into a lower domain space will certainly sacrifice some information. However, as shown in the results, it does not sacrifice as much as one would think, at least with this dataset. The variance in results was also rather surprising. One would expect that the results would vary a tremendous amount simply because the projections are randomized. However, even through an average of 5, the results did not deviate more than 5% from the overall average both for KMeans and EM. This appears to be a testament to the amount of information that can be gained from each and every attribute. Overall, both KMeans and EM ended with a k of 2 being the best with the best error metrics being roughly 15% and 17% respectively which is not significantly worse than any of the other algorithms which were used.

Wine Dataset:

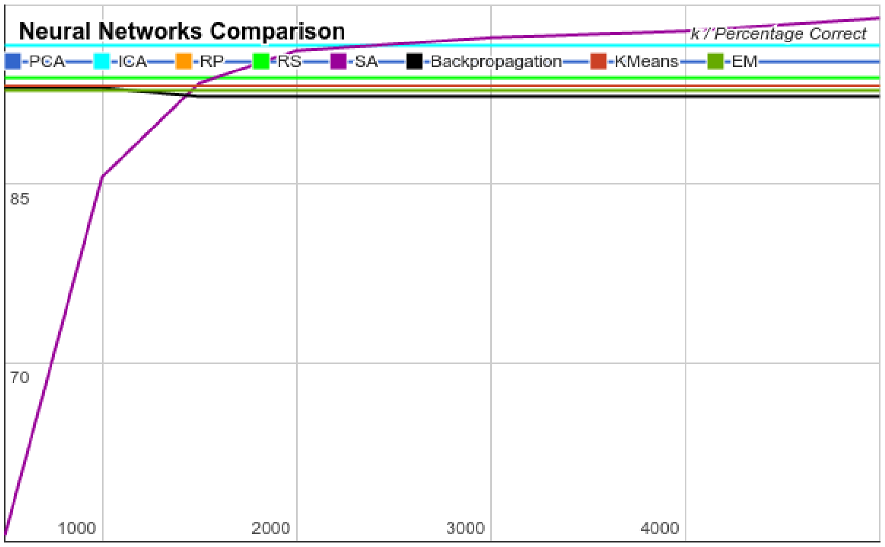
This dataset yielded similar results to the vote dataset. Randomized Projections did not end up being as bad as I thought it would be initially. For KMeans, it certainly produced the worst results while it produced reasonable results for EM. That being said, the results it produced for KMeans weren't all that bad and the averages across multiple trials were relatively consistent, typically staying within 7% of the overall mean. The best result that it produced was when k was 2, with an error rate of roughly 53% which in fact was right around what was produced by the default KMeans algorithm. It certainly did not improve as k increased and in fact was never even close to the other algorithms from there on out. Certainly information was lost but it clearly was not as much we thought initially. However the amount of information lost became more and more evident as k increased showing us how much more powerful PCA and ICA truly are. Randomized Projections didn't do horribly for EM however, at least not as poorly as it did for KMeans. I think this is due to the fact that the data was still spread out by lowering the dimensions even though it may have been random. There certainly was information which was lost but evidently the information that was lost actually did not end up hurting the overall result as much as one would think initially. It still failed spectacularly after crossing into larger values of k but for lower values of k, it did alright. For each value of k, no value deviated from the average by more than 10%. Information was certainly lost but this only began to matter was the values of k became larger.

RS:

Out of all of the algorithms run, this one produced perhaps the most surprising results. Randomness can often cause issues but in some situations, it can be a wonderful thing. At a glance, one would expect RS to perform very poorly given that it just removes arbitrary amounts of data, some of which could be very important. But in both datasets, this algorithm showed that removing data which does not contribute much to the overall class attribute may in fact reduce complexity thereby making clustering and classifying easier. RS did not provide an incredible amount of help when it came to KMeans. It did not perform incredibly well though it did more or less do as well as the other algorithms. For the vote dataset, the lowest error was 13.41% with a k of 2 which is right around the value that default KMeans returned (and is actually better than the default value for PCA). For the wine dataset, the best result was an error rate of 51.77% with a k of 2 which is better than default KMeans as well as RP.

Where RS performed extremely well was when it was combined with EM which was surprising considering randomness usually leads to issues in problems like this. The fact that it was so consistent over an average of 5 per value of k is incredible. I believe this is a function of the fact that all attributes had a relatively similar information gain. Members of congress typically vote along party lines while wine quality is not necessarily a function of the number of attributes which were present. When RS was combined with EM, I believe it essentially worked as if it had removed a tremendous amount of noise from the dataset. It lowered the complexity, essentially turning a problem with a pretty large number of attributes into a problem with only 10 attributes. Naturally there is an element of randomness but clearly doing this helped. Specifically, I believe that this helped when running EM because the datasets became a little more spread out. By this I mean, things that related them together in higher dimensions were removed thereby making the sets more distinct. As a result, the soft clustering actually worked well because very few points fell in the gray area created by the probability distributions. This concept makes sense too given that the results were so consistent even when the value of k was changed. The fact that the values became more distinct helped EM when clustering. While EM paired with RS still did not yield the best overall results, it certainly was the most consistent and that alone makes it worth discussing. For the vote dataset, the average error percentage was 29.320% across all values of k. With the wine dataset, the average percentage of assignments which were incorrect was 60.3427%. Both of these numbers are incredibly low given the range of k values which were used. In terms of individual results, the best value produced by EM using RS in the vote dataset was an error rate of 14.2361% with a k of 2. This value is better than default EM, RP, or even default PCA which is worth noting. The value for the wine dataset which was the most intriguing was the error rate of 52.7792% when k was set to 3. This is astounding because it is the second best value in all of EM for this dataset just behind ICA with a k of 2. It seems that there was a considerable amount of noise removed by RS after all..

Neural Networks:



Looking at the Neural Network results above (Training results have been omitted as they all hit 100% accuracy within the first 1000 iterations with the exception of Simulated Annealing, which was discussed in the previous assignment), Simulated Annealing still ends up being the overall best algorithm. It is also worth noting that Randomized Projections and the Neural Network using the EM as added features produced the exact same accuracies both for training and testing accuracy. For the purposes of PCA and ICA, the algorithms that yielded the best results on the vote dataset above were used in the neural network with the expectation that more information in that regard would help with classification. In addition, when running all of the algorithms as well as the clustering algorithms to add as features, a value of 2 was used for k since that performed the best in every instance above.

Analyzing the results, the neural network performed pretty much as one would expect. Simulated Annealing still won out over everything. In terms of the dimensionality reduction algorithms, ICA won by a reasonable margin which correlates directly with the fact that ICA produced the overall best results during clustering. PCA-5 worked the second best which again correlates with the fact that it very slightly beat out the default KMeans results on the vote dataset while clustering. RS worked best after PCA-5 which shows that each attribute has a reasonable amount of information gain. Reducing a random number of attributes will lower accuracy as information is lost but the fact that it still performed this well shows that there is a lot of information left for the neural network to learn from. KMeans itself worked the next best and then RP and EM tied each other. This is interesting as it still follows the trend that KMeans was better than EM in every instance involving the vote dataset.

EM itself was no better than randomizing projections on the dataset which shows some pretty poor performance. That being said, even with this poor performance, each and every algorithm that was used in this assignment along with Simulated Annealing from the last assignment performed better than backpropagation. In addition, none of them lowered their accuracy at all as the number of iterations increased showing that they didn’t overfit while backpropagation overfit slightly. I believe the reason for this is that the data which is provided to the neural network via these algorithms is inherently flawed. There is no algorithm which produced a 100% accuracy on clustering and as a result, adding this information to the neural network, regardless of the way that it was added, gives the network a different set of information which is still very accurate albeit in a different way. In fact, the dimensionality reduction algorithms simply restructure the information and do not remove anything unless it is negligible. So, the information is presented in a way which is more conducive to learning because the data is more spread out. It’s clearly still not perfect otherwise clustering algorithms would result in no error for reasonable values of k. The fact that the data is still flawed, just restructured is the reason that it performs better. There is less of an overlap in the data and there is a direct correlation between how algorithms performed for clustering above and how they performed here. In addition, the fact that the algorithms which performed well effectively remove noise when implemented correctly means that overfitting is less likely as there is nothing to overfit to. Even the clusters helped with accuracy over simple backpropagation. This unique bit of information allows the neural network to have another input which does not directly coincide with the class attribute but coincides quite a bit. This leads to increased learning while not allowing it to overfit because the neural network never receives any form of clustering which it could learn too much from.