

Unsupervised Learning Assignment

Yuanheng Wang
902905501

Dataset Description:

winequality-white. This dataset consists of instances of Vinho Verde white wine. The task is to predict the quality of the wine, represented as an integer between 0 and 10. There are 4,898 instances and 12 attributes, including class. Each attribute is an objective, numeric value (such as pH), whereas the quality is a subjective, nominal value. The data is visualized on the right, with a quality of “3” on the far left going to “9” on the far right. This data is close to a normal distribution.

King-Rook vs King-Pawn. This dataset is a description of end game in chess, where white has a rook and king left and black has a pawn and king left. The class is a binary result of whether white can win or not. There are 36 attributes that describe the board and the last one is the classification: “win” or “nowin” and 3196 instances.

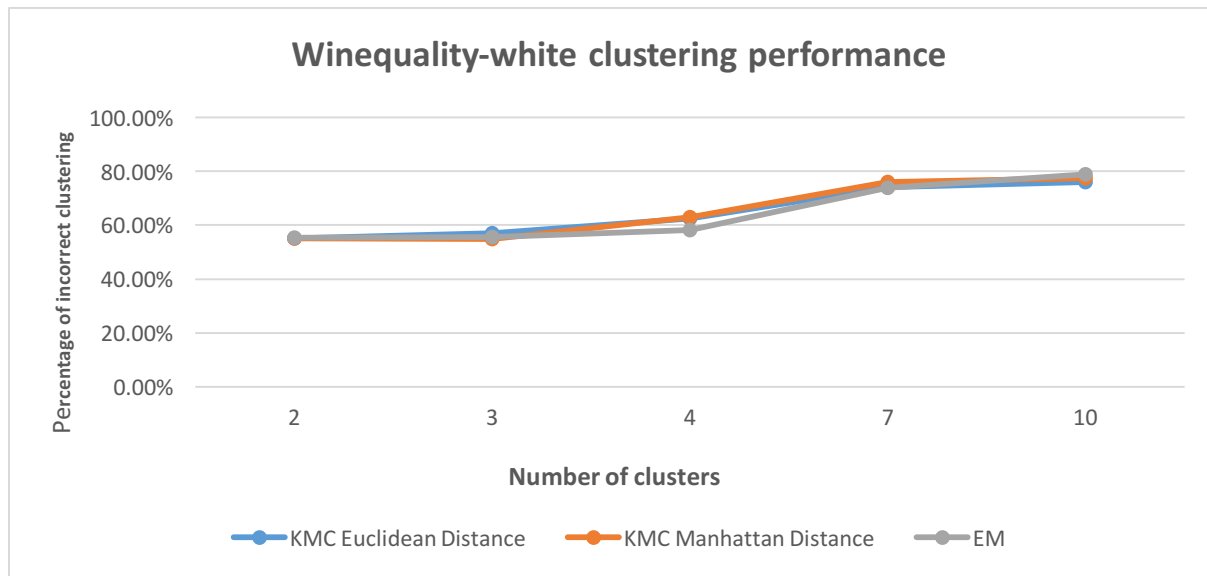
Using these two dataset, we can compare the results of multivariate labels versus binary labels as well as attributes who have multiple values versus attributes that only have two representations.

K-Means Clustering and Expected Maximization:

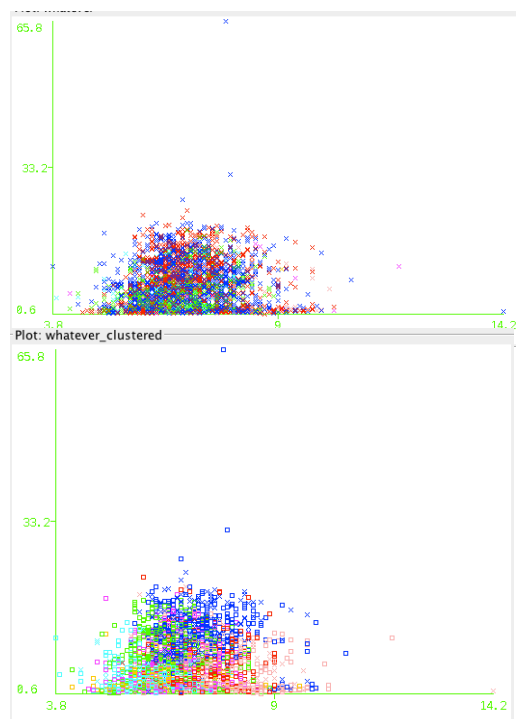
To run K-Means Clustering and Expected Maximization algorithms, I used Weka developer version. The seed is always set to 10 to make randomized algorithm like KMC to generate repeatable results.

Winequality-white problem:

Below are the performance graphs after running the two algorithms. The maxIterations are set as 500, and seed number is 10 for the purpose of consistency. I tested out different cluster numbers in order to identify certain trends as clustering number differs but I specifically analyze the performance when the amount of clusters is the same as actual classes to compare percentage of incorrectly clustered instances.

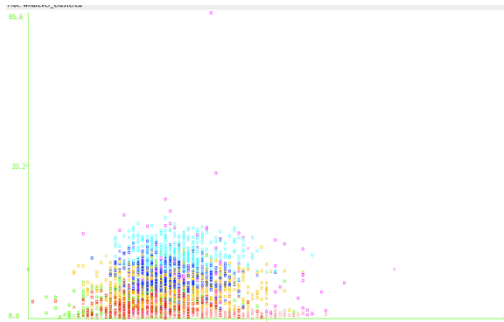


The clustering algorithm doesn't seem to work that well for either for the dataset. For whine data, the original thought I had was percentage of error will be the lowest when there are 7 clusters, which corresponds to 7 qualities from 3-9. In reality, the percentage of clustering error rises as number of clusters increase. Looking into the assignment matrix, the distribution of clusters is very similar to the true distribution of classes, but among the clusters, only 20 to 30 percent is correctly clustered. For example, for KMC, only 25 percent of quality at "6" are correctly grouped and 34 percent of quality "5" classes are correctly clustered. EM also displays very similar behaviors. This is very likely caused by the skews of the dataset itself. With 12 attributes, this high dimensionality can make distance measures such like Euclidean distance and Manhattan distance ineffective.



By comparing the visualizations of true classes vs clusters after applying algorithms, we realize the complexity of the problem. We randomly pick two attributes, and set "Fixed Acidity" as x axis and "residual sugar" as y axis. The graphs on the left, from top to bottom are: true classes, KMC result and EM result. Each color represents different groups. We can tell that the original data has a lot of overlapping, which adds in great difficulty of algorithm application.

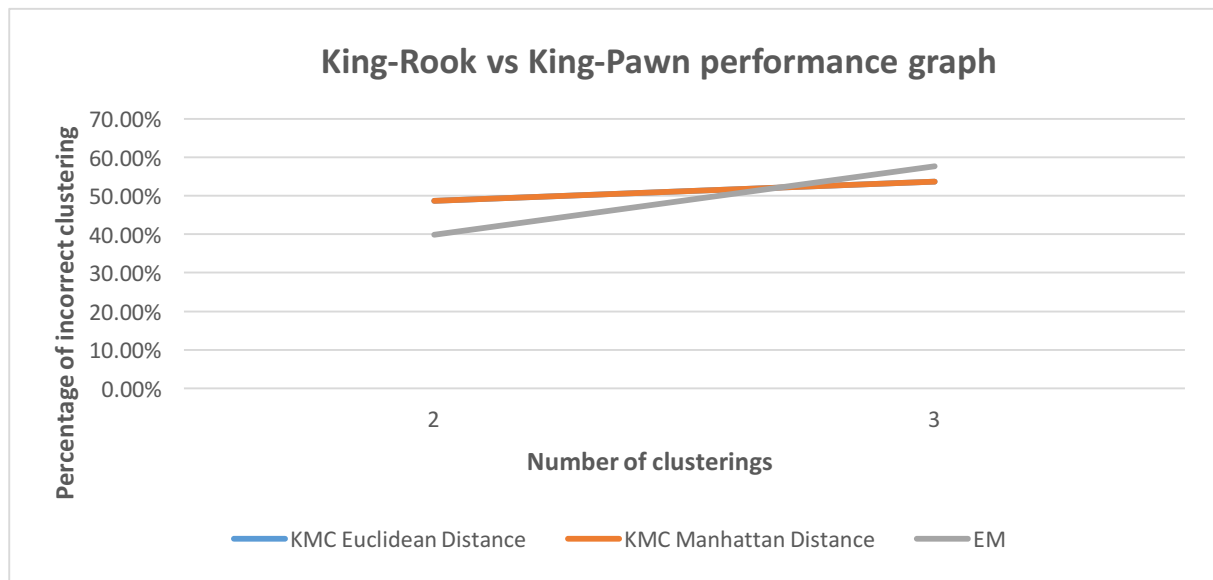
In terms of difference between KMC and EM, we can find that EM has a clearer bound, although the actual clustering accuracy is pretty close with KMC. The reason is probably due to the mechanism that EM takes soft clustering the those on the border between two clusters contributes more information. Overall, the poor performance and the similar results between two algorithms suggest that the data is probably non-



globular (or non-spherical). As we know, KMC and EM are very similar, but KMC holds the bias that clusters are round/spherical and solid where EM doesn't assume the shape of the data. If it's a spherical data KMC is supposed to do better.

Chess problem:

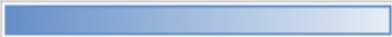


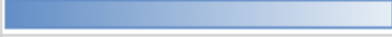







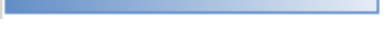
As mentioned, the result for this dataset wasn't great either, but we see a 10 percent improvement in terms of accuracy for EM than K-Means Clustering.















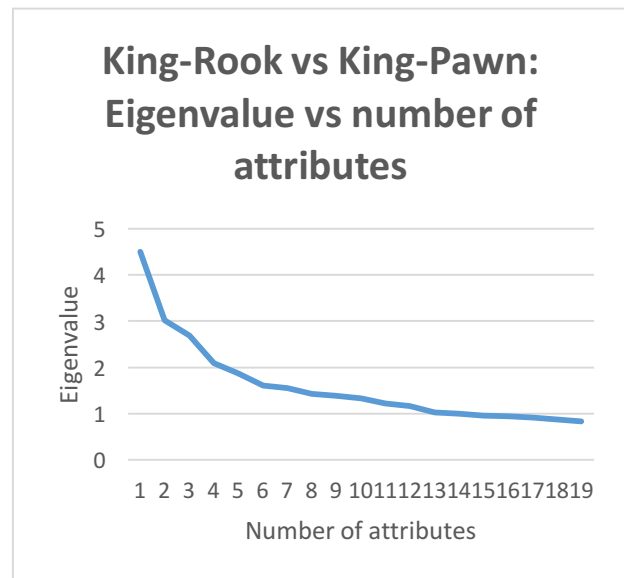
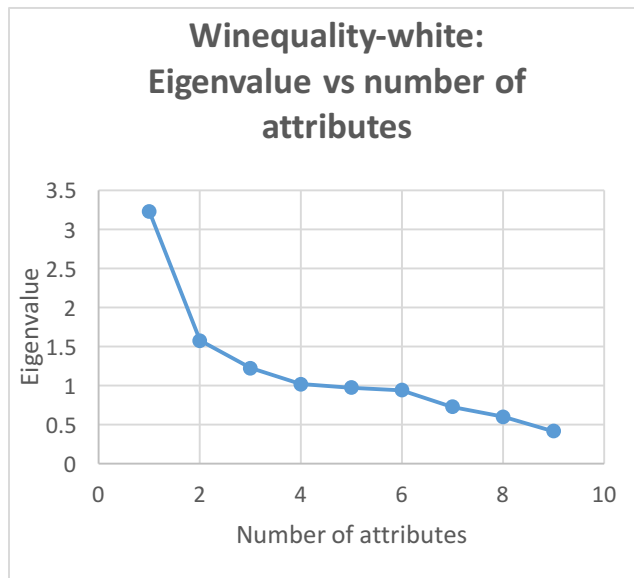
Although there is only two classes, but the main reason of of the inaccuracy likely stems from the many attributes (36). In addition, all the attributes have binary values, that means either that description of chess board is true or false. These two properties together make the “distance” between final output really small and very hard to cluster. On the other hand, thinking back to project 1 where we applied classification supervise learning algorithms, the accuracies are very high for all the algorithms. This is a great example to show the difference between how supervised learning and unsupervised learning works and how invaluable training information can be to improve accuracy.

Principle Component Analysis:

Below are the running results after applying PCA dimensionality reduction. The clusters are 7 for wine data and 2 for chess data.

Winequality-White				
Algorithm	Threshold	Attributes selected	Incorrectly Clustered	
KMC Euclidean	0.95	9		74.98%
KMC Euclidean	0.9	8		73.13%
KMC Euclidean	0.8	6		75.64%
KMC Euclidean	0.5	3		75.52%
KMC Manhattan	0.95	9		76.40%
KMC Manhattan	0.9	8		77.38%
KMC Manhattan	0.8	6		77.36%
KMC Manhattan	0.5	3		76.99%
EM	0.95	9		76.40%
EM	0.9	8		68.62%
EM	0.8	6		67.19%
EM	0.5	3		72.65%

King-Rook vs King-Pawn				
Algorithm	Threshold	Attributes selected	Incorrectly Clustered	
KMC Euclidean	0.95	31		48.39%
KMC Euclidean	0.9	27		48.30%
KMC Euclidean	0.8	21		48.26%
KMC Euclidean	0.5	9		48.26%
KMC Manhattan	0.95	31		48.30%
KMC Manhattan	0.9	27		48.53%
KMC Manhattan	0.8	21		47.59%
KMC Manhattan	0.5	9		48.08%
EM	0.95	31		40.52%
EM	0.9	27		42.98%
EM	0.8	21		44.10%
EM	0.5	9		49.87%



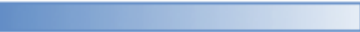







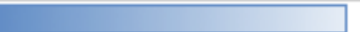
For the wine dataset, comparing the result with the raw data clustering from part 1, there is no big difference between rate of correct clustering for KMC. For EM, when the threshold (or variance covered) is around 0.9 and 0.8, there is an improvement over accuracy. This is probably that removal of some “noisy” attributes that confused the probabilistic distribution previously. After certain amount of attributes reduction, however, the PCA algorithm starts taking out the attributes that actually contribute the correct information and error rate climbs again.










For the chess problem, attributes selection doesn’t seem to enhance the performance. On the other hand, for EM algorithm, the percentage of incorrectly cluster grows constantly when I keep reducing the number of attributes I selected. This means all the attributes contribute. I verified the trueness of the claims by looking at the stdDev(standard deviation) data at the statistic panel as well as the ranked attributes. There I found that rankings decrease slowly with no attributes that by itself contains majority of the information. By looking at the correlation matrix, it’s also showing that attributes have little correlation, usually bouncing between ± 0.1 .

In order to verify the right choice of attributes, I plotted eigenvalue against the number of new attributes we want to keep. For wine data, it seems the first elbow appears at 2. When using only two attributes, which accounts for 40 percent of the variance, we got an inaccuracy rate of 70.01%, which is worse than using 6 attributes. For chess data, there doesn’t seem to be a particular elbow but rather fairly smooth curve. This is consistent with including majority of the attributes. Overall, scree test seems to be more subjective and arguable of what is the best number of attributes to keep. Hence, we will determine the amount of attributes that come with the highest accuracy rate. For wine dataset, we pick 6 attributes and for chess, we include 31 attributes.

Independent Component Analysis:

First of all, I ran ICA and after generating independent attributes, I ran different algorithms. After that, I calculated the kurtosis of each attributes and select those that represent most non-Gaussian form distributions. Below are the performance results after running the KMC and EM for each scenario.

Winequality-White				
Kurtosis	Algorithms	attributes	Incorrectly Clustered	
N/A	KMC Euclidean	11		70.33%
	KMC Manhattan	11		73.84%
	EM	11		72.93%
Kurtosis > 1	KMC Euclidean	8		75.81%
	KMC Manhattan	8		76.45%
	EM	8		69.54%
Kurtosis > 1.5	KMC Euclidean	5		73.99%
	KMC Manhattan	5		76.84%
	EM	5		67.37%

King-Rook vs King-Pawn				
Kurtosis	Algorithms	attributes	Incorrectly Clustered	
N/A	KMC Euclidean	36		14.79%
	KMC Manhattan	36		12.33%
	EM	36		39.68%
Kurtosis > 1	KMC Euclidean	30		19.37%
	KMC Manhattan	30		22.92%
	EM	30		41.59%
Kurtosis > 1.5	KMC Euclidean	18		23.68%
	KMC Manhattan	18		21.40%
	EM	18		41.95%

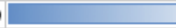
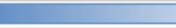
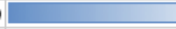
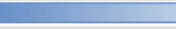














Looking at the outcome for wine data, there is only a tiny improvement for using ICA determined independent attributes when all of the attributes are included for clustering. That means although there is some independency, it's not strong. A wine can be rated good only it has the right mixture of majority of attributes comprehensively.



















For chess problem, I had to apply NominalToBinary filter before applying ICA because the ICA in WEKA only applies to numeric value. What's surprising is that "magic" happens after I clustered using the new attributes. KMC algorithm had a substantial augmentation. Even after taking out half of the attributes, KMC still made the incorrect clustering rate as low as 20 percent, where EM still performs around the same. My hypothesis is that ICA well separate the features that are independent of each other, for example, maybe some chessboard description metrics are completely independent and unrelated. The data sort of got transformed to a more globular

form. Then taking out the attributes that don't contribute much information will have only minimal effect. For EM, since it uses soft clustering, it still considers points on the edges of two clusters and that's why there is no big change in accuracy.

Randomized Projections

For randomized projection, I used Gaussian as distribution to calculate matrix and applied different seed on different number of attributes to account for the random nature of the algorithm. Below are the results of running. An interesting change is that attributes correlation plot are all stretched to form linear and Gaussian shape.

Winequality-White					
Algorithms	attributes	Incorrectly Clustered seed = 42		Incorrectly Clustered seed = 60	
KMC Euclidean	10		76.99%		76.50%
KMC Manhattan	10		78.17%		78.06%
EM	10		76.85%		74.56%
KMC Euclidean	8		77.40%		76.60%
KMC Manhattan	8		79.67%		79.34%
EM	8		77.58%		77.89%
KMC Euclidean	5		77.09%		77.68%
KMC Manhattan	5		79.15%		80.67%
EM	5		77.36%		78.22%

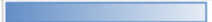
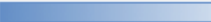







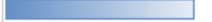
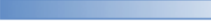
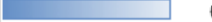






King-Rook vs King-Pawn					
Algorithms	attributes	Incorrectly Clustered seed = 42		Incorrectly Clustered seed = 60	
KMC Euclidean	25		47.59%		47.99%
KMC Manhattan	25		47.54%		47.72%
EM	25		45.84%		48.79%
KMC Euclidean	15		47.69%		49.01%
KMC Manhattan	15		49.02%		50.00%
EM	15		49.33%		49.24%
KMC Euclidean	10		49.24%		50.00%
KMC Manhattan	10		46.74%		49.46%
EM	10		47.57%		49.62%


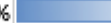

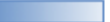
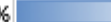













RP does well on reconstructing the dataset as the results are pretty similar to those using original raw data attributes. Exclusion of less important new attributes also doesn't affect the result too much.

For the wine dataset, the change of seed doesn't change the outcome too much, but for chess dataset, results do fluctuate significantly upon various seed numbers. When there are 25 attributes and 10 attributes, algorithms with seed number 80 does more than 10 percent better in terms of clustering accuracy. This is very interesting as different seed number is used to generate random matrix. If the result varies so much, that means the randomness actually changed the distribution when projecting high dimension attributes to lower dimensions. This is likely due to the lack of data due to the curse of dimensionality as more attributes need more data to support. This is why the wine dataset has less variance regarding the final answer.

Randomized subsets:

I chose random subsets (RSS) as my final algorithm because I can use it as a baseline to compare the other reduction algorithms to. I tested it with a varying number of attributes as well as seeds, as before, because of the random nature of the algorithm. For the ease of analysis and what we realized from the previous runs, distance measure for KMC doesn't have notable effect due to the complexity of the space. Therefore, for RSS, I only show the answer of KMC with Euclidean distance.

Winequality-White				
Algorithms	attributes	Incorrectly Clustered seed = 42	Incorrectly Clustered seed = 60	Incorrectly Clustered seed = 80
KMC Euclidean	10	 74.54%	 73.81%	 73.95%
EM	10	 73.85%	 69.54%	 74.87%
KMC Euclidean	8	 77.66%	 74.64%	 70.87%
EM	8	 70.05%	 71.50%	 62.27%
KMC Euclidean	5	 74.83%	 73.00%	 71.80%
EM	5	 73.89%	 70.25%	 69.92%

King-Rook vs King-Pawn				
Algorithms	attributes	Incorrectly Clustered seed = 42	Incorrectly Clustered seed = 60	Incorrectly Clustered seed = 80
KMC Euclidean	25	 42.31%	 46.43%	 48.48%
EM	25	 39.81%	 48.66%	 39.23%
KMC Euclidean	15	 49.02%	 48.93%	 47.86%
EM	15	 39.63%	 48.66%	 49.02%
KMC Euclidean	10	 44.68%	 48.39%	 49.51%
EM	10	 41.73%	 46.43%	 47.41%

Overall, wine's clustering performance are more consistent (except for EM with 8 attributes and seed = 80). Again, for chess clustering, we found several cells with incorrect clustering dropped underneath 40 percent. This is similar to RP algorithm we ran previously. This confirms my idea that for randomized algorithm like RP and RSS, when the dataset is not large enough, there contain certain combination of attributes and randomized seed number that will render the best clustering result.

General Comments about dimensionality reduction:


































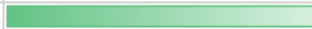

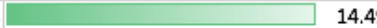














Overall, the performance of KMC and EM after having at least some attributes removed seem to be worse than the case when KMC and EM are applied without dimensionality reduction. But tuning the composition of certain parameters in the algorithm can create attributes that contain the most mutual information with original attributes and reconstructs back to the original model.

Something I can do in the future to improve the results is to have a data with relatively less attributes or more instances. This way, by applying random algorithms I won't get too distinct outcome. In reality, however, this can be really difficult unless the original data contain many information less attributes, which rarely happens. One potential dataset I can think of applying is crime data where the attributes of a crime, according to my research experience can be over 200. Many of those attributes seem useless even with bare eyes. But to find the attributes that

contains most information, a systematic method like what I've been doing for this homework can definitely apply.

Running Neural Networks:

To see the effects of dimension reduction algorithms on neural networks, I ran each dimensionality reduction algorithm on the wine dataset. (I chose wine over heart because I have existing NN data from assignment 1.) Then I ran Multilayer Perceptron on the new data, with the standard parameters of 0.3 learning rate, 0.2 momentum, 500 epochs, and 10-fold cross validation, seeking to classify based on wine quality. The results are in the table below. The graph underneath was the original NN results from project 2 just for the purpose of reference.

Winequality-white			
Reduction	Clusters	Incorrect classified Instances	Time (seconds)
Original	None	 44.77%	 11.25
PCA, variance = 0.95	None	 44.39%	 9.72
PCA, variance = 0.8	None	 47.04%	 7.24
PCA, variance = 0.5	None	 52.04%	 5.53
PCA, variance = 0.95	KMC	 7.92%	 15.72
PCA, variance = 0.8	KMC	 7.68%	 13.63
PCA, variance = 0.5	KMC	 6.31%	 9.84
PCA, variance = 0.95	EM	 11.90%	 15.56
PCA, variance = 0.8	EM	 7.37%	 13.01
PCA, variance = 0.5	EM	 8.11%	 10.12
ICA	None	 43.69%	 10.93
ICA	KMC	 9.45%	 17.88
ICA	EM	 10.23%	 16.86
RP, attributes = 10, seed = 60	None	 49.49%	 9.84
RP, attributes = 8, seed = 60	None	 51.25%	 8.49
RP, attributes = 10, seed = 60	KMC	 4.92%	 16.54
RP, attributes = 8, seed = 60	KMC	 5.45%	 14.29
RP, attributes = 10, seed = 60	EM	 4.57%	 16.46
RP, attributes = 8, seed = 60	EM	 6.06%	 14.49
RSS, attributes = 10, seed = 60	None	 44.14%	 9.55
RSS, attributes = 8 seed = 60	None	 49.44%	 8.44
RSS, attributes = 10, seed = 60	KMC	 5.72%	 16.32
RSS, attributes = 8 seed = 60	KMC	 6.27%	 15.52
RSS, attributes = 10, seed = 60	EM	 12.33%	 16.76
RSS, attributes = 8 seed = 60	EM	 14.19%	 14.61

In general, NNs on the reduced dataset produced similar, slightly better (PCA with 0.95 variance covered and ICA with 11 attributes) or worse results when compared to plain NNs from assignment 1. And from the results we can tell that after reducing attributes to certain level, reducing it more will decrease the prediction accuracy. This means that the original attributes

all contribute information to make it more predictable with classification. By removing attributes and data, we actually filtered out the useful information.

Nonetheless, the results are significantly different when performing dimensionality reduction but also applying clustering. In general, it seems that classification accuracy actually improved a lot with the inclusion of cluster data. The inaccuracy dropped to as low as 5 percent! I suspect that, given the noisy data, the inclusion of these clusters helps to reduce the noise. The most likely reason the clusters still help is because the clusters are grouping similar instances together. Although these clusters do not correspond to the actual quality classification, the extra information the clusters provide still help 'guide' the NN as it is updating weights.

With more attributes, RP and RSS both saw slightly better results. This is sensible for the wine dataset. The inclusion of the 'guiding' cluster attribute makes it more likely for these randomized algorithms to find a 'good hit'. I think it makes sense that KMC's NN performance would be better than EM's or without clustering's. Because it hard- assigns clusters, more information is captured in KMC than in EM or without clustering at all. This allows the NN to take advantage of the extra information gained through clustering.

In terms of speed, we find that theory holds true that the less attributes and data are included, the faster running neural network becomes. When running the ANN using original attributes with all the data, it took the longest to complete. So in real life when we are dealing with data, we need to consider the tradeoff between runtime and the amount of information we can keep from original raw data.

Conclusion:

Dimension Reduction is definitely a useful technique that we can fully employ when dealing with big data. It's going to work well when we know some existing attributes contributes noisy or little information with the support of many instances. By integrating clustering technique, it will help to improve classification accuracy dramatically with the right choice of clusters. Next when applying dimension reduction, we definitely need to think carefully and choose the algorithm that works the best. When all the other algorithms don't work well, RP and RSS are experimentally useful and can be played around with.