Datasets Description

Car Evaluation Database

This dataset was derived from a simple hierarchical decision model. All attributes in this dataset are nominal values, and there are 4 labels/classes of car acceptabilities: unacc (unacceptable), acc (acceptable), good and vgood (very good).

This dataset is interesting because it is based on a given and known hierarchical decision model. All attributes have semantic meanings embedded with their classes. It is simple for a human to say that a car with low price, and high comfort and safety must be very good. A car with high price, and low comfort and safety must be unacceptable. However, with many different levels of lows and highs given for each attribute, the dataset is way more complex than it seems. It would be interesting to see how this dataset behaves with unsupervised algorithms.

White Wine Quality Dataset

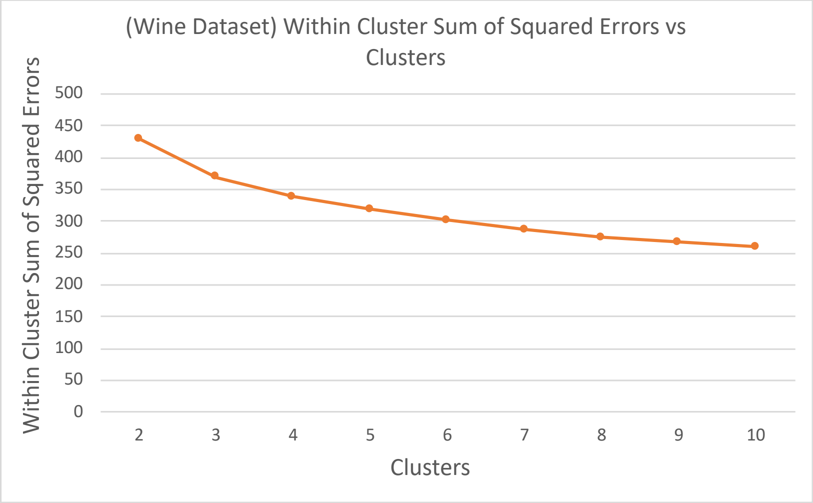
The inputs/attributes are objective tests, numeric values (e.g. PH values, fixed acidity). The outputs/labels are classes of 'good' and 'not good' based on evaluations by wine experts.

In contrast with the cars quality dataset, the attributes don't make much semantic sense with the classes. It is almost impossible for anyone to infer the quality of a wine given all of its objective test values. The quality of a wine is a very subjective indication: different experts might have very different evaluations on wine quality. The test results are very objective numeric values. This property of this dataset makes it extremely interesting but also complex. In previous assignments, this dataset is concluded to form a difficult classification problem and a complex problem space. Therefore, this dataset is also used for the neural networks section on this assignment.

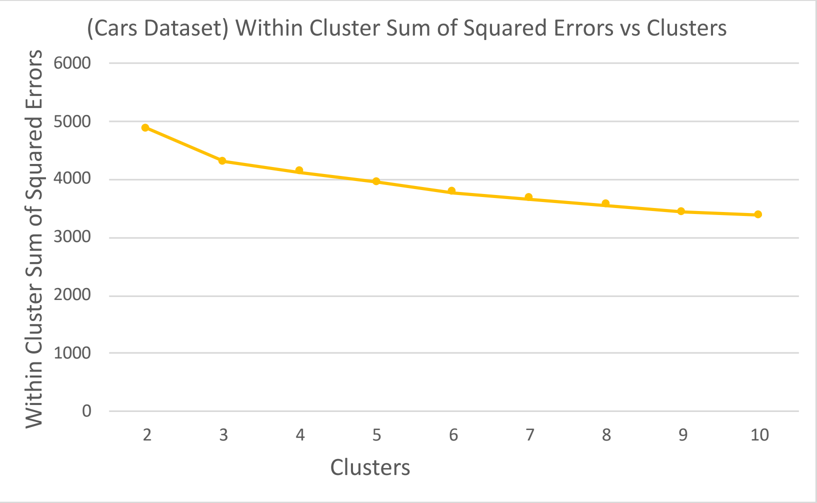
Unsupervised Learning (Clustering & Dimensionality Reduction)

Clustering - K-Means

K centroids are initially scattered on the problem space, and each centroid acts as a center of a cluster, resulting in k clusters. All data points are then assigned to the closest centroids. Then, the centroids adjust themselves to the centers of the clusters. According to the new centroids positions, the cluster assignments are performed again for all data points. Such procedure repeats until convergence (no more changes to the centroids and cluster assignment).

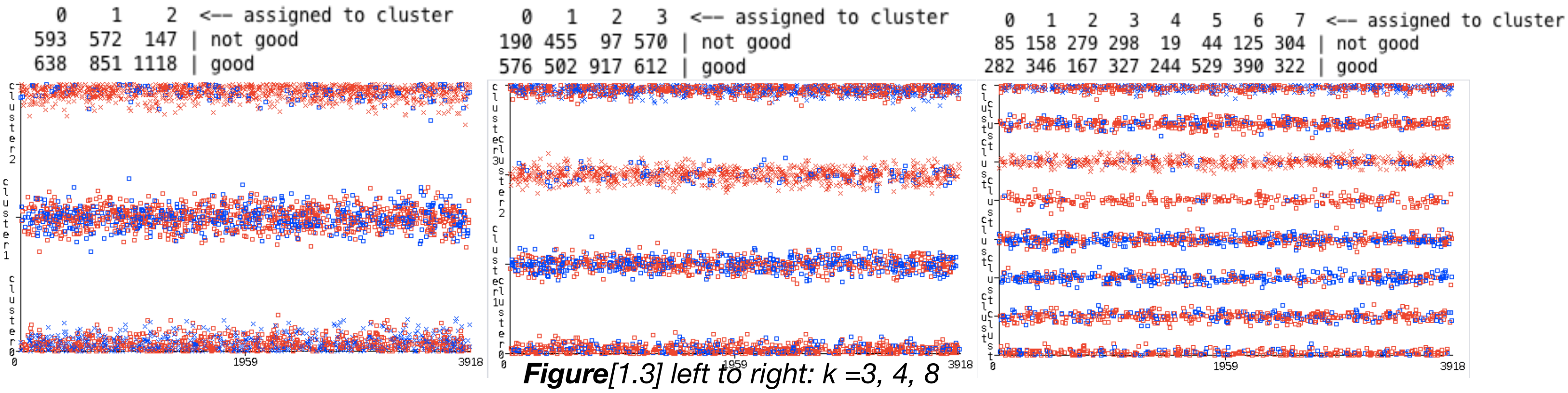


Figure*[1.1]*



Figure*[1.2]*

Figure[1.1] is the sum of squared errors vs number of clusters graph. The curve has a trend of decreasing while the number of clusters increases, as expected (theoretically, sum of squared error goes to zero when k equals the number of instances). However, the smoothness of the curve is unexpected according to the elbow method. The elbow method mentions that for the first few additional clusters, the sum of squared errors would drop rapidly, and the drop slows down at a certain point, forming a noticeable angle in the curve (an elbow in an arm). That point of switch would mark the appropriate k value. However, in Figure[1.1], such a point is hard to find. This indicates that this dataset is naturally not clustered, which, from previous assignments, is actually expected. It is previously concluded that this dataset constructs a complex problem space and this graph supports this conclusion by providing that this dataset doesn't have naturally clustered groups/classes. From closer looking at the graph and numerical data, clusters of 3, 4, and 8, before which there seem to be slightly more rapid drops, can be good candidate k values for now.



```
  0    1    2   <-- assigned to cluster
 593  572  147 | not good
 638  851 1118 | good
```

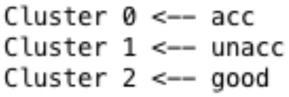
```
  0   1   2   3   <-- assigned to cluster
190 455  97 570 | not good
576 502 917 612 | good
```

```
  0   1   2   3   4   5   6   7   <-- assigned to cluster
 85 158 279 298  19  44 125 304 | not good
282 346 167 327 244 529 390 322 | good
```

***Figure**[1.3] left to right: k =3, 4, 8*

Figure[1.3] shows further analysis on this dataset. The y-axis is the clusters and the colors are the two classes (red="good", blue="not good"). When k=3, cluster 2 captures a fair portion of the "good"(red) class, but the other two clusters are still very mixed. When k =4, nothing seems to improve. When k=8, visually, there are obviously two rows of blue and two rows of red in the middle of the graph. Unlike previous clusterings, cluster 2 in this one actually captures more of the "not good"(blue) class, and most of the clusters seem to contain a dominating class, which makes each cluster less ambiguous in terms of classes. Since in this dataset, overfitting is the least worry (neural network's best training performance is always at 70%-80%. There is a complex problem space to fit), I would choose the bigger k, k=8, which actually makes semantic sense as well because this dataset was preprocessed from a dataset of 11 classes (scores of 0-10) with some empty classes, so **k=8 seems very reasonable** and aligns back to the original dataset.

Figure[1.2] is the same graph as Figure[1.1] for the cars dataset. This one has a slightly more obvious elbow at k=3. This elbow demonstrates that the clusterings are gaining less information from additional clusters after k=3 than before k=3. k=3 also makes sense because this dataset has 4 classes with few instances in the "very good" class. Thus the dataset reasonably clusters into three main clusters of other three classes. Figure[1.4] illustrates exactly what is expected for the 3 clusters. Therefore, based on both the elbow method and the prior knowledge of this dataset, **k=3 would be appropriate for k-means on this dataset**. The elbow is still a little too smooth to make a definite conclusion, which again suggests that this dataset might not cluster well. This



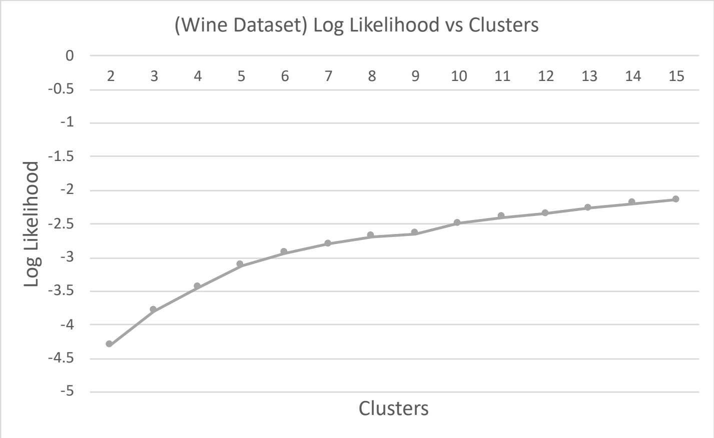
```
Cluster 0 <-- acc
Cluster 1 <-- unacc
Cluster 2 <-- good
```

***Figure**[1.4]*
Analysis from Weka

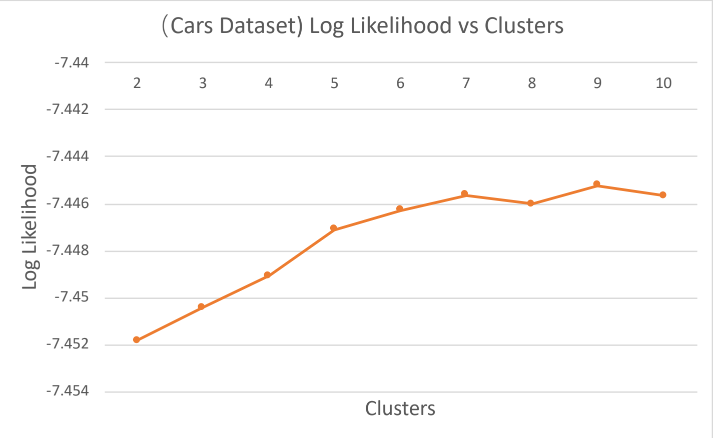
might be due to the unbalanced classes in the dataset (almost 70% of the data is “unacceptable”, and 3%-4% of the data is “good” or “very good”).

Clustering - GMM-EM

Unlike k-means, GMM-EM doesn’t assign data points into clusters, instead, GMM-EM uses soft clusterings to give each data point probabilities of belonging to each cluster. K gaussian models are initialized with random parameters, and the initial log likelihood is computed for the data points space. For the E step, evaluate the responsibilities for all data points using the current parameters. For the M step, reestimate the parameters with current responsibilities. Evaluate the log likelihood again, and repeat the EM steps until convergence (no change in log likelihood). Similar to k-means, process of assigning clusters (responsibilities) and adjusting centroids (reestimate parameters) repeat until convergence.

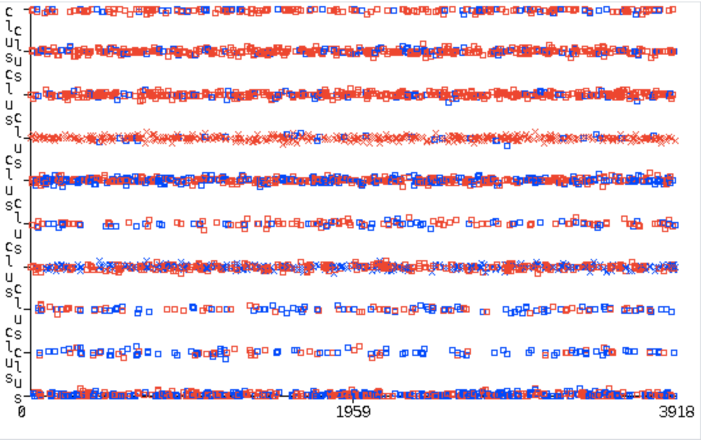


***Figure**[2.1]*

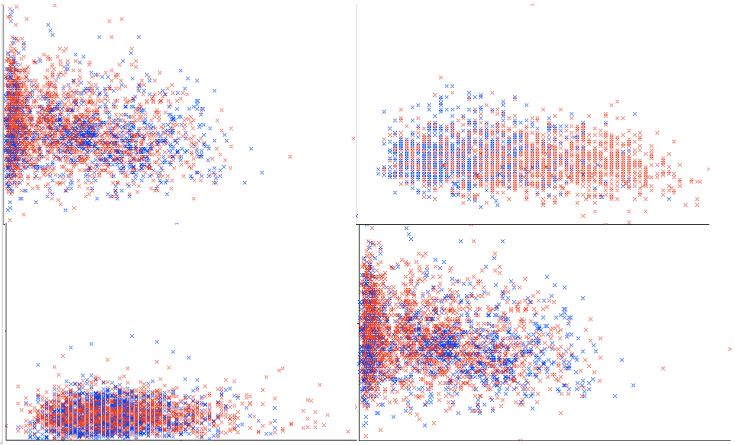


***Figure**[2.2]*

Figure[2.1] is the log likelihood vs number of clusters graph, which is very consistent with Figure[1.1] in terms of the smoothness. However, since GMM-EM allows soft assignment, which allows overlapping clusters and ambiguous cluster assignment, it can capture more nuances in the problem space, so the curve shows that, even at 10 clusters, there is still more information gained than expected. Therefore up to 15 clusters are tested to obtain more knowledge on the behavior. From the behavior of the curve, 10 clusters seem to be an interesting entry point for analysis.



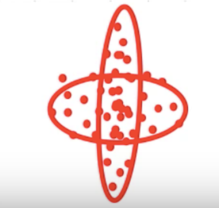
***Figure**[2.3] Clustering Visualization*



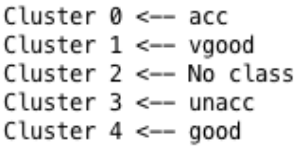
***Figure**[2.4] Classes/labels Visualization*

Figure[2.3] and Figure[2.4] are some further analysis. From previous assignments, this dataset is known to be hard to fit. Figure[2.4] shows some original dataset visualization given random two features, which are mostly extremely mixed. Based on this observation, in a bigger picture, the dataset must have the two classes extremely mixed together as well, which gives factual evidence to my analysis on the dataset being complex in nature. The upper right visualization in Figure[2.4] actually appear to have the two classes separated pretty nicely. This shows that this problem is still solvable, which makes it interesting. Figure[2.3] shows the class distributions in the 10 clusters. Visually, there are some red clusters, blue clusters and of course some mixed ones. Given the complexity of the problem space, this clustering result is already great at k=10. Therefore **k=10 would be appropriate** for GMM-EM on this dataset for now. An even bigger k might produce even better clusterings since overfitting is not an issue.

In Figure[2.2], there seems to be two obvious elbows: one at k=5 and one at k=7. However, the one at 5 is preferred because the curve seems to be just oscillating from k=6 to 10 at around the same level, suggesting a flattening overall trend. **Thus 5 clusters seem to be the better option**. k=3 was determined to be the most appropriate for k-means on this dataset, but GMM-EM seems to keep benefitting more from cluster numbers of 4 and 5, which makes perfect sense. This behavior can be explained with the underlying difference between k-means and GMM-EM. For the data points in Figure[2.5], two natural groups are obvious to human eyes, but k-means can never cluster these groups; whereas, this is an easy clustering for GMM-EM. The cars dataset might be similar to this. <u>Given all the combinations of the attributes, if one attribute is bad (for example, low safety), no matter how good other attributes are, this instance is probably “unacceptable”, resulting in many possibilities for “unacceptable” class. However, there are not many ways to have “very good” ones because this class requires very particular combinations (for example, reasonable price, safety, etc.), just one very good attribute doesn’t make this instance “very good”. This also explains the dominating number of “unacceptable” instances. Based on this analysis, the “unacceptable” or “acceptable” data must be scattered everywhere, and the “good” or “very good” data clustered in the problem space</u>. This problem space is almost like a more complex version of Figure[2.5], where there exists ambiguous regions: overlapped (“unacceptable” ones are everywhere) but also clustered in a reasonable way (“very good” ones). Therefore, GMM-EM reasonably keep benefitting from the space beyond k-means’ limit because GMM-EM seems more suitable for this problem. k=5 also makes perfect sense in this dataset. 4 of the clusters would be the 4 classes, and the last cluster would be the noise/outliers. To back up the analysis, figure[2.6] is a label/class assignment to the clusters by Weka.



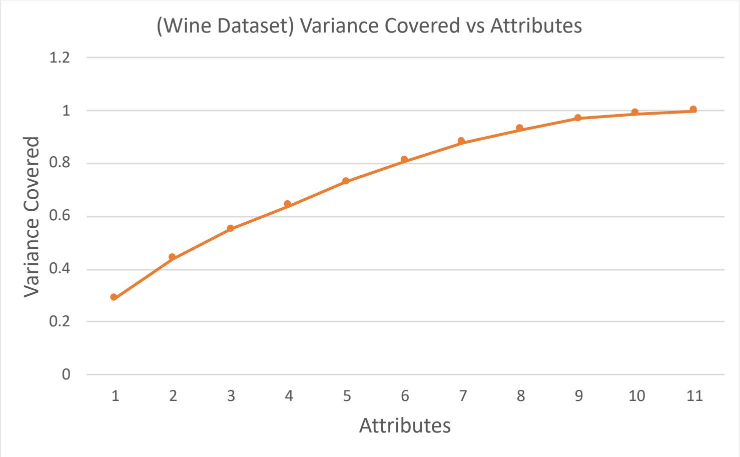
Figure*[2.5]*



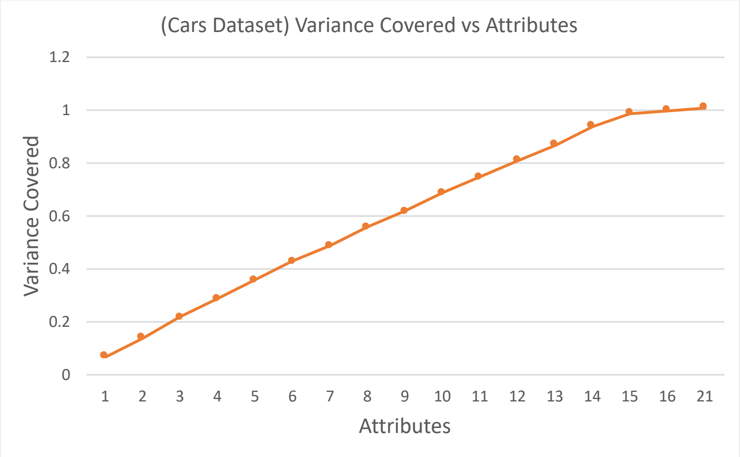
```
Cluster 0 <-- acc
Cluster 1 <-- vgood
Cluster 2 <-- No class
Cluster 3 <-- unacc
Cluster 4 <-- good
```

Figure*[2.6]*

Dimensionality Reduction - PCA



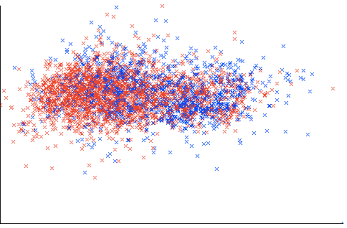
Figure*[3.1]*



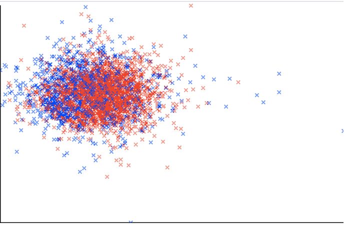
Figure*[3.2]*

PCA transforms the features of the dataset into a smaller set while maintaining the most information. This is achieved by taking the orthogonal projection of data onto a lower dimension linear space that maximizes the variance and minimizes the mean squared distance. This process starts off by taking a *d x d* covariance matrix on the current dataset, where *d* is the current dimension. Then eigenvalues and eigenvectors are computed. The eigenvectors are ranked based on their eigenvalues. The higher eigenvalues correspond to more important eigenvectors in terms of more variance. Each of these eigenvectors would correspond to a dimension in the target space.

Figure[3.1] is a variance covered vs number of attributes graph. This graph presents that, as expected, the variance covered decreases as the number of attributes decreases because more information is lost as less attributes are used. There is also an expected behavior of increasing rate of drop in variance covered as the attributes decrease. Essentially, when there are still many attributes, decreasing the number attributes won’t result in that much of a drop in variance covered; however, when there are not many attributes left, each decrement in the number of attributes results in a bigger decrement in the variance covered. This process can be dissected into two phases. The first phase is when there are still many attributes, where some attributes are nearly useless in terms of variance, so when PCA is performed, it is a process of optimizing the dataset by minimizing the dimensions while maintaining the information (putting less weight on the less variant features/dimensions). The second phase is when there are not many attributes left, and each attribute carries important information, PCA in this case is forcefully trimming dimensions at a risk of losing information. The optimal target dimension should be the transition point between the two phases, where the dataset is maximally optimized without much loss in information. If look closely to the curve in Figure[3.1], it behaves exactly like the explanation of 2 phases, and the transition point would be at 9 attributes, where there is a very slight bent in the curve marking a change in behavior. The variance is kept at 0.95 when there are 9 attributes, so **9 attributes would be a good target dimension as variance is still largely reserved**.



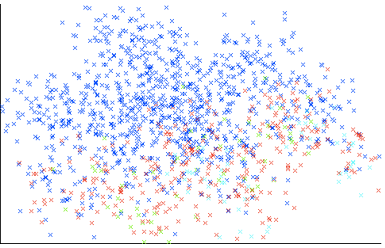
***Figure**[3.3.1]*



***Figure**[3.3.2]*

Figure[3.3.1] is a visualization for the two top ranked dimensions in the reduced space with 9 dimensions. The top ranked one is the x-axis, along which there is a clear separation of the two classes. However, the second ranked dimension (y-axis) doesn’t seem to separate the classes much. The eigenvalues are also consistent with this observation. The eigenvalue for the best ranked dimension is 3.29321 and the eigenvalue for the second ranked dimension dropped to 1.56317, and the rest of the eigenvalues never appeared to have this much of a drop. In this dataset, there is also a possibility of a dimension having a large variance, but not necessarily contribute to the classes because the documentation of this dataset also mentions that not all attributes of this dataset might be related to the classes. Therefore, this reduced space can provide insights to which attributes actually contribute to the classes. The top ranked dimension is “*0.509density-0.438alcohol+0.424residual sugar+…*”, which puts an emphasis on the attributes density, alcohol and residual sugar. This dimension is proven to be relevant in the visualization, so these attributes are more likely to be relevant. The second ranked dimension is “*0.582fixed acidity-0.579pH+ …*” (shown to not contribute much to classes in the visualization), which, by the same analysis, makes attributes fixed acidity and pH less relevant. Just to backup this analysis, Figure[3.3.2] has the second ranked dimension on the y-axis and the last ranked dimension on the x-axis. The visualization shows that the classes are still separated more along x-axis than y-axis. The last ranked dimension is "*0.629alcohol+0.448residual sugar+…*", which also emphasizes on alcohol and residual sugar just like the top ranked dimension. This supports the analysis that although there might not be much variance along the attributes alcohol and residual sugar, but they are at least relevant to the classes. Some other attributes, although having large variance in values, but are almost irrelevant to the classes, like fixed acidity and pH. For these reasons, PCA might not actually be an appropriate method to be directly applied to reduce many dimensions on this raw dataset (some relevant features might be trimmed away for not being variant enough). However, using PCA to simply optimize the dimensions without reducing dimensions is still a useful optimization (the top ranked dimension effectively separates classes). Based on the analysis, applying some feature selection method to select out the most appropriate features before applying PCA to reduce dimensions actually makes more sense.

Figure[3.2] is the same graph for the cars dataset, and the transition point is way more obvious. The original number of attributes is actually considered to be 21 because for discrete attributes, Weka considers each value a separate dimension. The variance can be kept at around 0.99 at attribute number of 15. That's 6 less dimensions with such a small loss in variance. This would be a good dimension for now because as a known characteristic of this dataset, each attribute here is proven to be extremely relevant (derived from a hierarchical decision model), and with each feature having different values, very different classes can be resulted, therefore it might be preferable to keep as most variance as possible. 15 attributes also mark the transition point between first phase and second phase. Therefore, **the point with 15 features seems to be a good target dimension**. Any less than 15 seems a bit too much variance/information lost for the dataset.

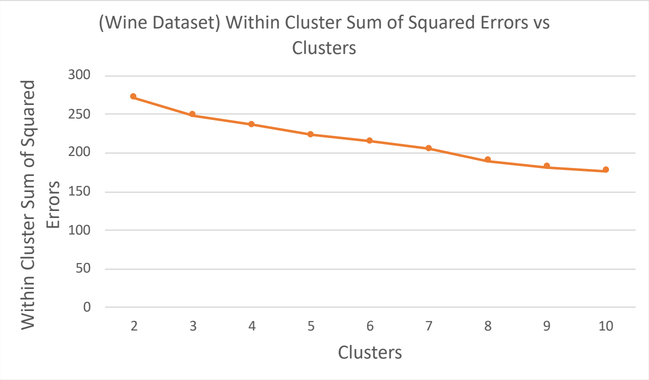


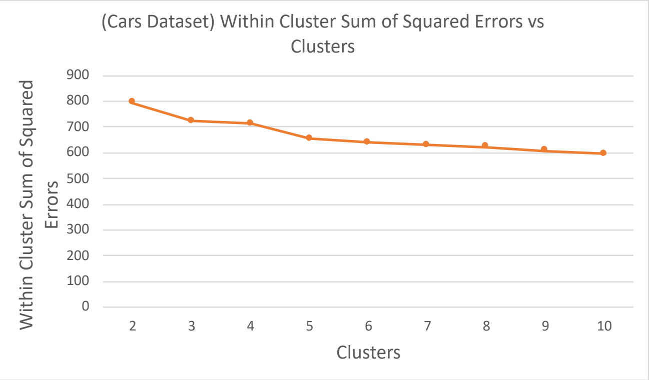
Figure*[3.4]*

Figure[3.4] is a visualization of the two top ranked dimensions in the resulting space. The y-axis does a great job separating the red and blue data points, and the x-axis is pretty good at separating the light blue and green points (zoom in to see clearer). However, the data still seems too mixed, especially the light blue or the greens ones with the red ones, which fully support the previous analysis on this dataset: *the "unaceptable"(blue) and "acceptable"(red) ones are everywhere, and the "good"(green) and "very good"(light blue) ones are more clustered*. *The analysis also mentions that many combinations of attributes result in blue or red points, but few combinations result in green or light blue points.* The green and light blue points also don't appear on the edge because they require a balanced and reasonable combination of attributes. Thus, this visualization of the two dimensions in figure[3.4] actually makes much sense and supports all my previous analysis. The eigenvalues also seem a lot more normal than the ones for the wine dataset. The first three are around 1.57678, 1.54874 and 1.53552, which are nicely ordered with similar intervals in between. The rest of the eigenvalues also follows this pattern.

The attribute numbers are chosen based on the previous analysis. Wine dataset is reduced to 9 attributes from 11 attributes and cars dataset is reduced to 15 attributes from 21 attributes.

K-Means After PCA

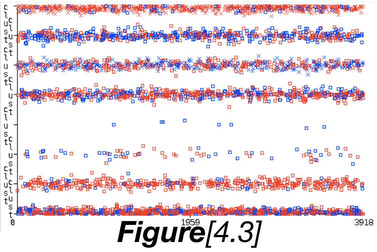




***Figure**[4.1]*

***Figure**[4.2]*

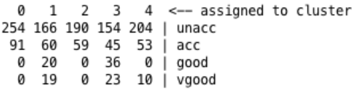
Figure[4.1] is the within cluster sum of squared errors vs number of clusters graph. The graph perfectly proves the analysis in the previous sections. The curve slightly shows that 3 and 8 are possible elbow positions, which are consistent with Figure[1.1], and they are showing in a much more obvious manner than Figure[1.1]. This observation strongly supports the choice of the target dimension, which effectively optimized the dataset into lower dimensions with the most information maintained, and thus supporting the analysis about PCA's 2 phases. Comparing to Figure[1.1], the elbow at 3 clusters is smoothened and the elbow at 8 clusters is magnified. This observation supports that 8 would be a better choice for the k value, and 8 is exactly the cluster number selected for k-means on the wine dataset previously as well. **After the PCA reduction to 9 attributes, k is also selected to be at 8** for the same reasons as Figure[1.1] with stronger evidence in Figure[4.1]



***Figure**[4.3]*

Figure[4.3] is the visualization for such clusterings. There are some red clusters, some blue ones and some mixed ones just like before. However, although cluster 3 only has 8 instances, they are all blue. For the k-means on the raw version of this dataset, the blue ("not good" class) ones are found to be extremely difficult to cluster out. Based on this insight, this visualization supports that the dataset is effectively optimized.

Figure[4.2] is the same graph on the cars dataset. The curve shows two elbows at 3 and 5 clusters. The elbow at 3 clusters is consistent with Figure[1.2], which is the version before PCA. However, the elbow at 5 clusters actually is consistent with the one in Figure[2.2], which is the GMM-EM clustering on the raw cars dataset. In that part of the analysis, the theory is that the dataset has overlapping clusters that only GMM-EM could cluster. This observation shows that the PCA dimensionality reduction actually effectively reduced the overlapping dimensions and highlighted the separation of data. Thus, **k-means can also gain sufficient information at 5 clusters after the PCA reduction to 15 attributes**.

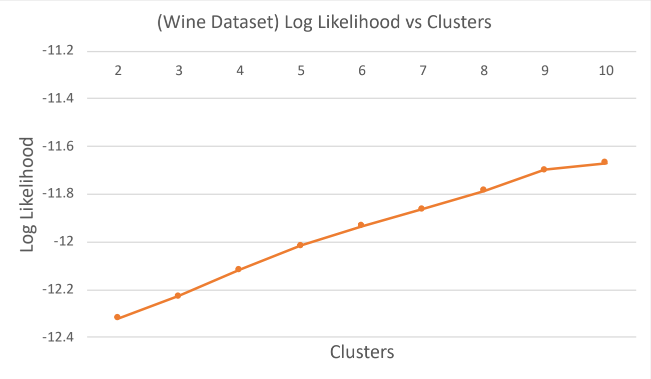


```
  0   1   2   3   4  <-- assigned to cluster
254 166 190 154 204 | unacc
 91  60  59  45  53 | acc
  0  20   0  36   0 | good
  0  19   0  23  10 | vgood
```

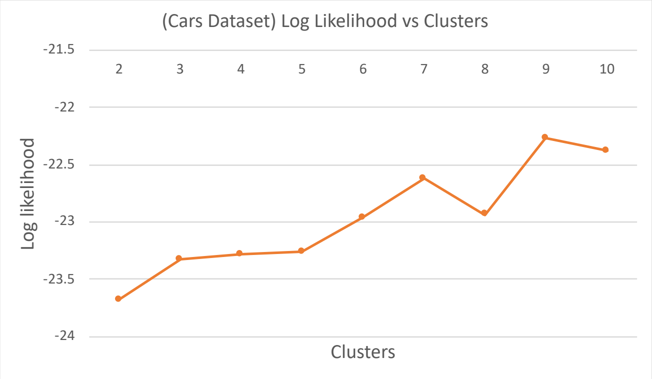
***Figure**[4.4]*

Figure[4.4] shows the cluster assignments. Most data points are still largely overlapped, which is an underlying nature of the dataset. Figure[4.4] also directly supports the analysis that the "unacceptable" and "acceptable" data points are scattered everywhere causing overlaps (high numbers in all clusters), but the “good” and “very good” classes are more clustered (high numbers in few clusters).

GMM-EM After PCA

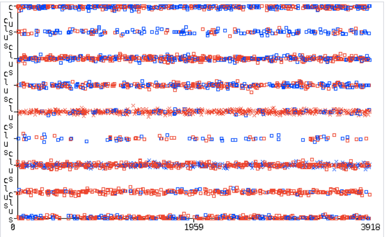


Figure[5.1]



Figure[5.2]

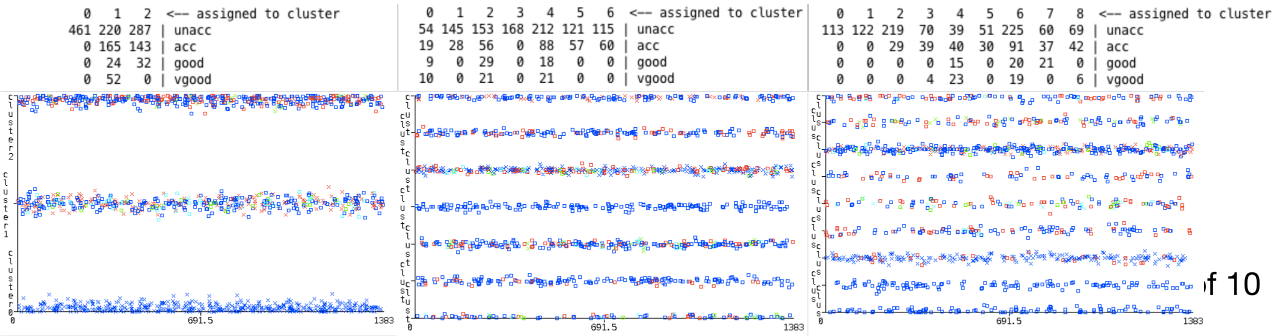
Figure[5.1] is the log likelihood vs number of clusters graph. The overall trend is consistent in the curve, which makes the elbow at k=9 pretty obvious. This information, although different from Figure[2.1], still makes perfect sense. In Figure[2.1], k=10 is selected to be appropriate for GMM-EM on the raw dataset. With the PCA dimensionality reduction, the dataset is projected onto a lower dimensional space while 0.95 of the variance is preserved. This process can be understood as an optimization of the dataset attributes. With the optimization, the dataset is reasonably relatively less complex comparing to before. The elbow at 9 shows that the same can be achieved on the new space with less clusters needed than before. The decrease in the appropriate k value supports that this problem space is effectively reduced.



Figure[5.3]

Figure[5.3] is a visualization of the clusters at k=9 on this reduced space. Visually, there are some blue rows on the top, some red rows and some mixed ones just like the ones obtained from k=10 on the raw dataset (Figure[2.3]), which supports the analysis that similar clusterings are achieved with less clusters.

Figure[5.2] is the same graph for the cars dataset. This graph highlights the k values of 3,7 and 9, before which a rapid increase appears. This k value of 3 here corresponds back to k=5 in Figure[2.2] because with the dataset optimized, less clusterings are reasonably needed to gain similar information. k=7 and 9 are actually ignored in Figure[2.2] because they just seem like oscillating around the same value, suggesting a flattening trend, and the curve here in Figure[5.2] actually appears to have a similar behavior. However, in Figure[5.2], they don’t seem to oscillate around the same level any more, so a further analysis might be needed on this behavior.



```
  0   1   2  <-- assigned to cluster
461 220 287 | unacc
  0 165 143 | acc
  0  24  32 | good
  0  52   0 | vgood
```

```
 0   1   2   3   4   5   6  <-- assigned to cluster
54 145 153 168 212 121 115 | unacc
19  28  56   0  88  57  60 | acc
 9   0  29   0  18   0   0 | good
10   0  21   0  21   0   0 | vgood
```

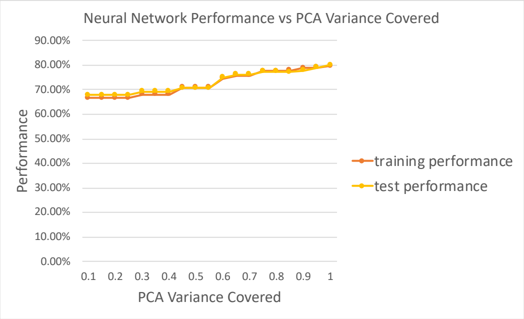
```
  0   1   2   3   4   5   6   7   8  <-- assigned to cluster
113 122 219  70  39  51 225  60  69 | unacc
  0   0  29  39  40  30  91  37  42 | acc
  0   0   0   0  15   0  20  21   0 | good
  0   0   0   4  23   0  19   0   6 | vgood
```

Figure[5.4] *left to right: k =3, 7, 9*

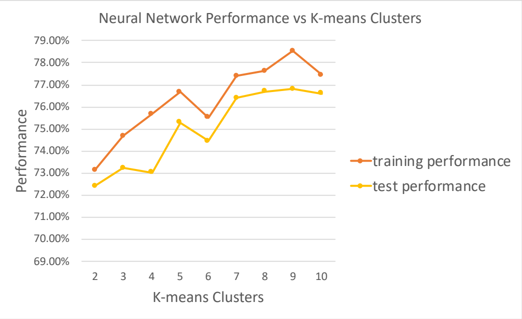
Figure[5.4] contains the clustering information and visualization for these interesting k values. When k=3, the clusterings are very interesting. Almost half of the “unacc” ones are clustered into one cluster exclusively. All of the “vgood” ones are clustered into the same cluster as well. These observations might suggest a favorable clustering at k=3, but because of the nature of the dataset (too many “unacc” instances scattered everywhere) all the clusters are essentially still dominated by the “unacc” class. As for k=9, it is using more clusters to separate all the “unacc” ones, and hopefully, achieve a cluster dominated by a minority class. From the visualization, there are actually some clusters that seem less blue (“unacc”). Before this, blue points just seem to be dominating all clusters. However, while separating all the “unacc” ones, the “vgood” and “good” ones are also separated. Therefore, reaching to a cluster that has dominating minority classes might require an unreasonable large k value. For this reason, **cluster of 3 is actually an appropriate k value** because it can at least achieve an all “unacc” cluster with such small k value and a cluster that contains all the “vgood” ones. Based on all these experiments and the nature of this dataset, clustering might not actually be suitable for this dataset because the “unacc” and “acc” ones seem to be too spread out.

Neural Networks

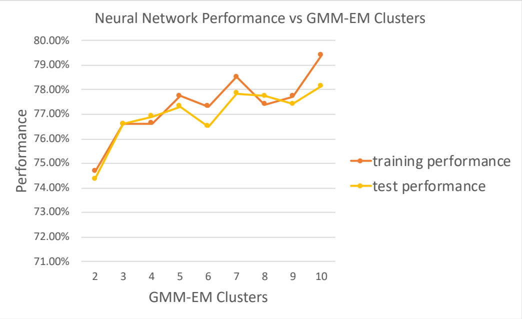
All of these processed datasets actually have faster training time than the raw dataset on neural networks because the attributes are summarized and simplified. PCA provides the best speed improvement.



***Figure**[6.1]*



***Figure**[6.2]*



***Figure**[6.3]*

Figure[6.1] is a performance vs PCA variance covered graph. The variance covered information is directly related to the number of dimensions in the result space: larger variance means more atirbutes, and variance of 1 basically means same dimension as before. This curve has an overall trend of increasing as the PCA variance covered increases. Basically, when the dimensionality is not reduced, the performance seems to be the highest, as soon as the dimensionality is reduced, even just by a little, the performance starts to drop. This observation provides factual evidence to supports the analysis from the PCA section: “*PCA might not actually be an appropriate method to directly reduce dimensions on this raw dataset. However, using PCA to optimize the dimensions without reducing many dimensions is still a useful optimization.*” As this curve shows, when the variance is kept at 1, where the dimensionality of the dataset remains unchanged (still 11 attributes), both of the test and training performances reach their peaks. The test performance actually got to the highest a neural networks have ever reached on this dataset, which is **80.08%**. The previous highest was achieved in the assignment 1, reaching a 78.14%. This accuracy improvement strongly supports the claim from earlier that *using PCA to optimize the problem space without reducing dimensions is still a useful optimization*.

Figure[6.2] and Figure[6.3] actually appear to have similar behaviors in the curves. They both have a lot of noisy behaviors in the curves, which might be largely due to the random nature of these two algorithms. However, they both have an increasing general overall trend as the number of cluster increases. This behavior supports the analysis that overfitting is the least worry in this dataset within this range of k values. As there are more clusters, the test performance seems to increase, which is consistent with how appropriate k values are selected for the clustering algorithms: larger k is preferred over smaller k. From the two test performance curves in Figure[6.3] and Figure[6.2], GMM-EM is also doing better than K-means on this dataset. The highest accuracy GMM-EM achieves is **78.14%**, whereas the highest accuracy K-means achieves is **76.61%**. This observation agrees with an analysis claimed earlier in the GMM-EM section: "*since GMM-EM allows soft assignment, which allows overlapping clusters and ambiguous cluster assignment, it can capture more nuances in the problem space*". Since the problem space for this dataset is extremely overlapped (as shown in Figure[2.4]), GMM-EM reasonably has a better result on this dataset than K-means.

From the performance of the neural networks on this dataset given all these different dataset preprocessing methods. PCA improves the dataset and the neural networks performance most effectively, achieving a high performance of **80.08%** on this dataset by neural networks. The clustering algorithms actually don't seem to improve the performance so much, which makes sense. From all the previous analysis on this dataset and problem space, the dataset just doesn't seem to naturally cluster into any classes. This is actually an important underlying nature of this dataset that makes it so difficult. The attributes of lab tests are not straight forward or intuitive evidences for the quality of the wine. Semantically, the quality of red wine actually has much to do with its drinking time temperature, container (even the shape of container), the duration that it is in contact with air, and so on. The lab results, by nature, affects the quality of the wine in a less significant degree than expected. This dataset, however, still provides an interesting problem space to solve because given certain lab tests of a particular wine, the probability of it being good or low quality can actually be predicted based on previous wine experience in the dataset, which is why 1-nearest neighbor was able to 96.22% on this problem. If directly looking at the problem space (Figure[2.4]) without the colorings of different classes, the problem space just looks like one giant cluster of data points. This suggests that clustering algorithms might not make much sense to be applied directly on this raw dataset, which provide explanation for the smoothness of the curves in Figure[1.1] and Figure[2.1]. It actually seems like what this dataset needs is some preprocessing on the attributes of the dataset, which might involve removing some useless attributes or adding some other more relevant attributes. The success with PCA, being a preprocessing of the dimensions and features, actually supports this claim of some preprocessing of the attributes and dimensions would be helpful.

One nature of applying PCA on this dataset should also be pointed out. Since all attributes are numeric values, in the perspective of PCA a change of 1.0 in values means more variance than a change of 0.1 in values. However, a 1.0 change in pH value might be a very trivial change in the context of pH value, and a 0.1 change in density might be a significant change in the context of density. Because all attributes have different contexts, just plainly compare the variance by the change in the numeric values seems unfair for some attributes. For future improvement of PCA on this dataset, the limit of variation for each attribute should also be taken into account to actually rank the dimensions in the most reasonable manner.

PCA achieve the highest test performance with 80.08%, also with the fastest training time of around 7 seconds.