Comparison and Analysis of Unsupervised Learning Algorithms
Sanjana Kadiveti

Datasets

In the following paper, I used the same datasets as I did in Assignment 1. A question I raised through a large part of the assignment is whether using clustering is a better idea for the datasets I have than supervised learning methods. My eventual response is, no, clustering is not a smarter idea for either of the datasets. With reasons discussed below, I analyze why these failures occur.

The Phishing dataset is a binary classification problem, and each of its 31 attributes has either 2 or 3 possible values. It has an overall of 11,055 instances, of which the distribution is fairly balanced: 55% positive and 45% negative. The point of the dataset is to classify whether a website is suspicious or not based on features such as, having an @ symbol, URL length, presence of redirects and pop-up windows, etc. I thought it was interesting because it seems simple in terms of the number of possible outcomes of both its attributes and its classification, but at the same time the dataset has a large number of attributes and instances. I wanted to explore the effect of this added complexity of size and dimensionality on the performance of these algorithms.

The Car dataset evaluates cars using 6 attributes: price, maintenance price, number of doors, boot size, capacity, and estimated safety, all of which are enum values, which I convert to numerical ones. The dataset has an overall of 1728 instances, and 4 possible classes of quality: unacceptable, acceptable, good and very good. I chose this dataset because it's simple and I was interested in how its simplicity would perform on unsupervised clustering. The data is already highly ordered, because each feature has four possible values, so in some sense the data is already quite tightly clustered. In addition, the dataset is unbalanced, skewed heavily towards the unacceptable and acceptable classes. On prior evaluation, one can judge that the dataset is patchy which suggests that linear methods such as PCA and Random Projection may perform poorly. I also wanted to compare performance between binary and multi-class classification problems, as well as between two datasets with different sizes and number of attributes.

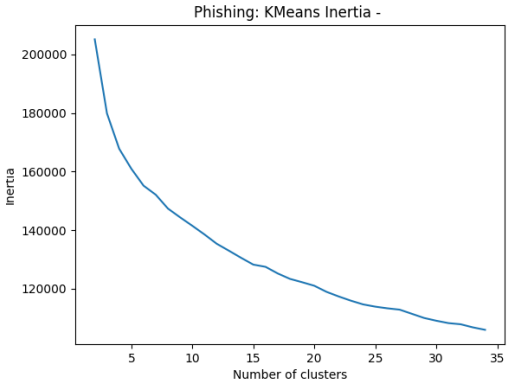
Clustering Algorithms

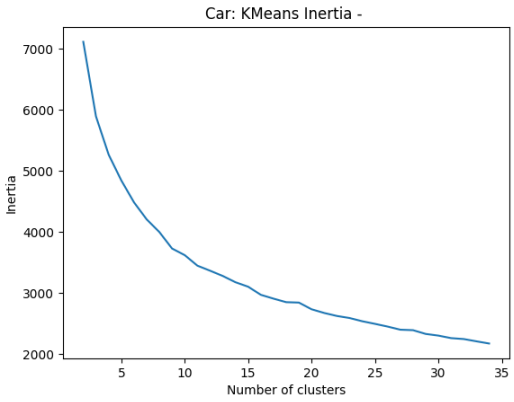
Clustering is used to find structure in unlabeled data by grouping together similar instances - the definition of similarity can differ between methods. We use K-Means clustering and Expectation maximization, which are both distance based clustering methods.

K-Means

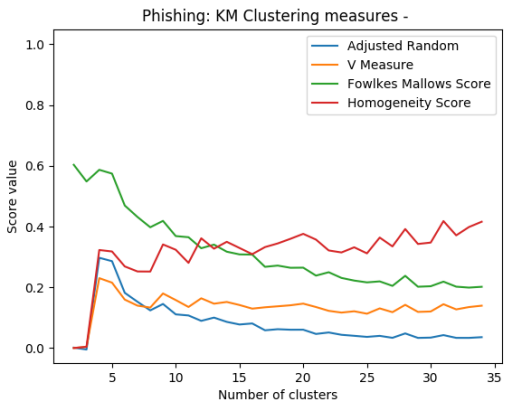
K-Means takes in k, the number of clusters to separate the data into, as an input and tries to find k clusters of equal variance by minimizing inertia, the sum of squared distances of the instances to the center of the closest cluster. More tightly compacted clusters have lower

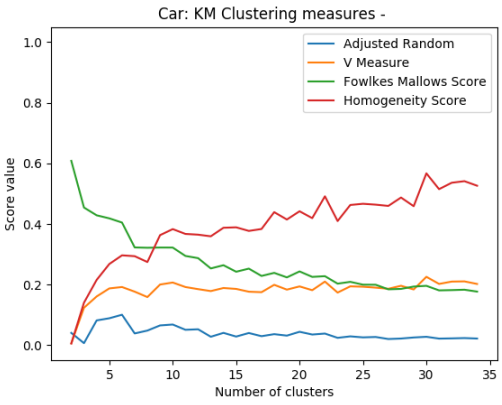
inertia than elongated ones. K-means first picks k samples randomly as centers of clusters, and then assigns all the other points in the dataset to its closest cluster center. The algorithm recalculates the best cluster center given the newly added points. The last two steps are then repeated until there is no change in inertia between iterations (the algorithm converges). A primary question to answer would be, how many clusters should we use for a dataset? To perform this experiment, I initially sweep over k values 2 to 34 for phishing and cars and plot inertia.





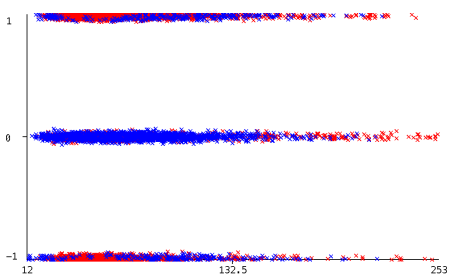
From the above graphs we see that as the number of clusters increases, sum of squared errors within clusters decrease. Using the elbow method is not so simple for either of the two datasets, but we do see more of an obvious point of curvature at k = 8 for the Car dataset. The phishing dataset on the other hand sees a smooth decline with no sharp elbows. The decrease in SSE doesn't represent accuracy however, since if we set k to be the number of instances, inertia would be 0. In search for other metrics to help disambiguate performance, I plotted the following graphs.

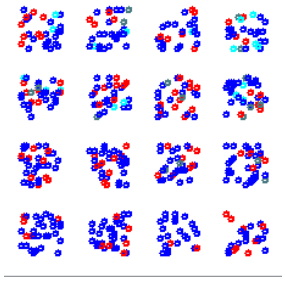




The figures show the performance of k-means using similarity metrics over different methods. Adjusted Random Index computes similarity between true labeling and predicted labeling. Homogeneity measures the general consensus within a cluster; a clustering satisfies homogeneity if all of its clusters contain data points that are members of the same class. High homogeneity scores mean the clusters separate based on ground truth labels. V-Measure is the harmonic mean between homogeneity and completeness, which is when all members of a class belong to the same cluster. Fowlkes Mallows is the geometric mean between precision and recall. We notice that the datasets have higher scores of Fowlkes

Mallows with few clusters; in particular, we notice that between 1 and 5 clusters, phishing doesn't see as much of a decline as Cars. Perhaps this is caused by the unequal distribution of class results in the Car dataset. The general decline in similarity scores is expected; nothing in either of the datasets suggest that a large number of clusters will be effective. Homogeneity is higher for more cluster in CE than in Phishing. Smaller clusters however have very low Homogeneity scores, which means the clusters don't line up with the true labels. We notice that for the Phishing dataset, the highest performance attained overall is with 4 clusters, while the Car dataset performs best at around 6 clusters. An interesting note to make is the fact that the Car dataset's homogeneity score sees a bump at k = 8, which is the k found using the elbow method. For this reason, I stick to 8 as my k for Cars. Since there was no K found using the elbow method with the phishing dataset, I used the similarity graph to decide on 4 for its k.



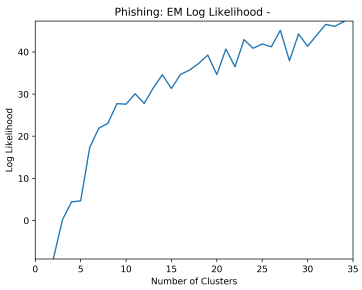


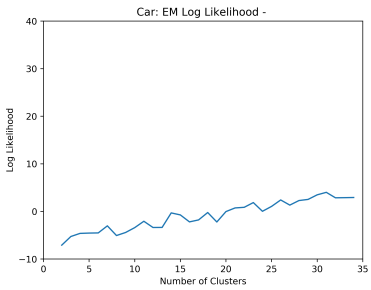
To the left are the WEKA clusters generated by K-means on Phishing and Car evaluation respectively. Plotting other attributes yielded similar results. At first glance we notice that the two produce very different yet equally messy clusters. The car dataset clusters form grid-like patterns because of their patchy, already clustered data, as mentioned in my introduction. Phishing on the other hand is less patchy, but contains many attributes that also have discrete values of 0, 1 or -1, which is why we still see separation over those values.

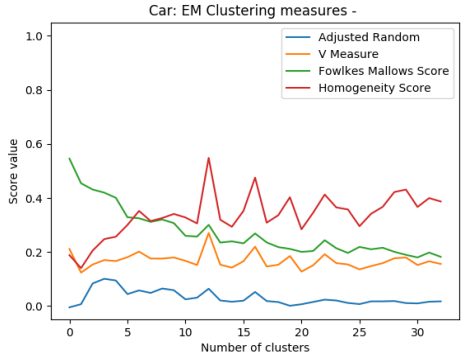
Error rates found with 8 and 4 clusters for Cars and Phishing respectively were 42% and 20.2% when running classification with just K-means. This is extremely high in comparison to other classification methods such as decision trees in assignment 1. (All of those methods reached a 90% accuracy and above for both datasets).

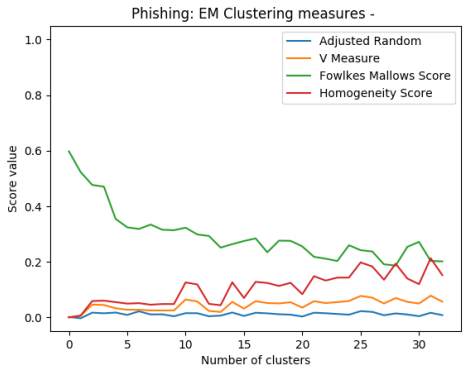
Expectation Maximization

This algorithm is a "soft" clustering algorithm, alternating between probabilistic expectation and maximization, and it assumes that all points are sampled from a number of Gaussian distributions. It finds the probability with which each point is generated by each of the Gaussian distributions and attempts to maximize the likelihood of obtaining the data given the distributions by altering its components' parameters.









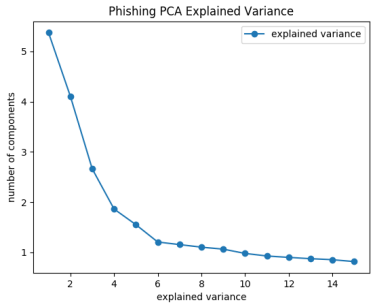
An interesting observation to note is that generally speaking, EM resulted in worse performance than K-Means. This may have been caused by inappropriate domain knowledge, such as outliers or imperfect data, or attributes that aren't very correlated to classification. The top two figures above display log likelihoods of the data as clusters increase. In both cases we see a general trend upwards, but the car dataset sees a smaller total increase than Phishing. This may have been because EM is vulnerable to getting stuck on local optima, and perhaps the Car dataset is a worse application of EM due to its uneven class distribution.
The left two figures show us performance in terms of the metrics described in the K-means section. Trend-wise, EM and KM show similar results. However, interestingly, homogeneity in Phishing is much lower with EM than with KM. In fact, every metric performs poorly with Phishing and EM. This could simply mean that Phishing doesn't demonstrate Gaussian properties and therefore cannot perform well with EM, and its clusters don't line up with true labels. There are other variations of k-means and EM methods in which centers are initialized differently which could have been used to achieve better clusterings.

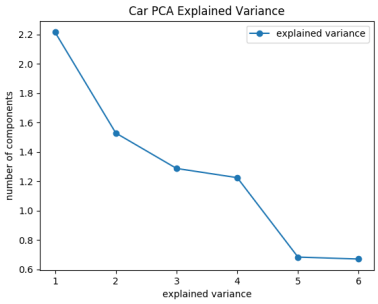
Dimensionality Reduction Algorithms

Often times, not all the attributes in a dataset contribute to classification. Additionally, the curse of dimensionality makes it preferred to have fewer features that describes the data well rather than the increased complexity of discovering how all the parameters relate to determine the most accurate outcome. Dimensionality reduction algorithms are used to find the most relevant features that determine classification.

Principal Component Analysis

PCA takes the target number of components as an input and uses eigenvectors and eigenvalues of a covariance matrix with n objects described by p variables to reduce the number of components to said target. Essentially, PCA tries to find the components that explains as much of the variance as possible.



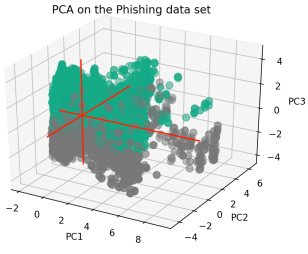


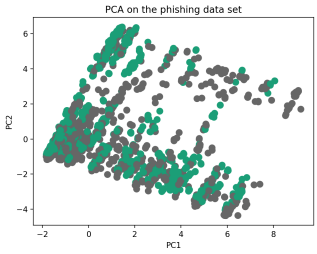
	Reconstruction Error
Phishing	0.44
Car Eval	0.24

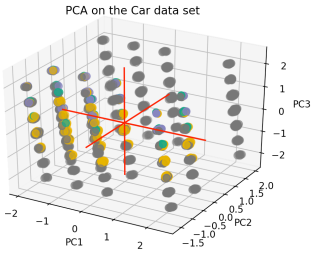
Note that the x and y axes labels for the graphs above are flipped – it should read explained variance vs number of components.

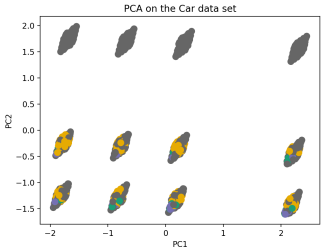
The graphs above show how much of the variance is explained by each eigenvector. For Phishing, the first 6 components capture most of its variance. This is excellent considering it has an overall of 31 attributes. The car dataset on the other hand, has 4 components that capture a great deal of the variance, and two that clearly contribute next to nothing in comparison, with an overall of 6 attributes. This demonstrates that PCA's distance metric is much more effective when applied on the Phishing dataset than on the Car dataset. Had I tested on different distance metrics, I could have found better performance results overall.

After finding the most relevant components, I ran PCA to reduce the dataset down to 6 and 4 components for phishing and car respectively. In the table above we see that reconstruction error is higher for Phishing with PCA than it is with Car Evaluation (CE). This is likely due to the fact that I reduced the number of features a lot more with Phishing than with CE. PCA creates linear combinations of less relevant features into a new feature set. This would create a lot of information loss for Phishing when reduced by so much.





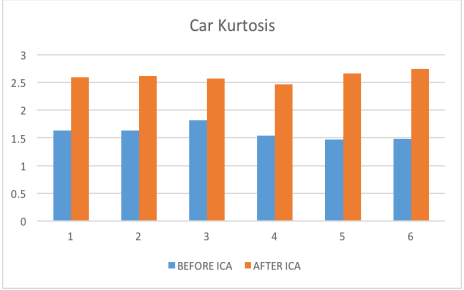


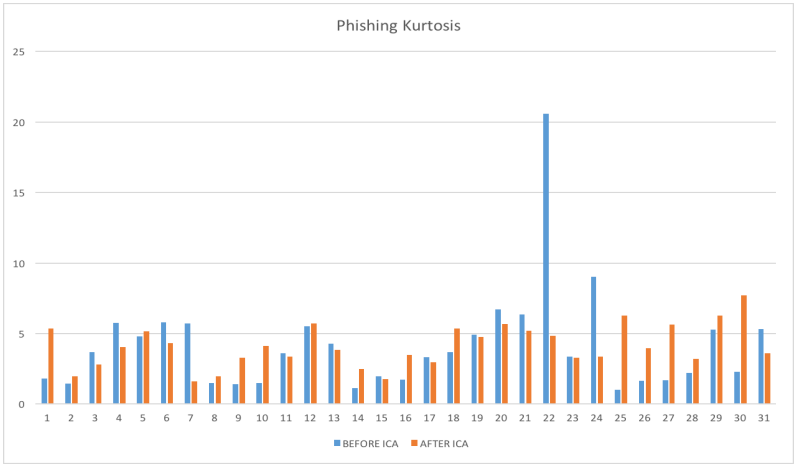


The graphs above visualize the projections of PCA on the phishing and car datasets in 3D and 2D space. There's a somewhat clear separation between the two classes in the Phishing dataset vertically, which we can see from the 3D plot. PCA doesn't separate phishing classes well on the other two components however, as we can see from the 2D plot. With the Car dataset in 3 dimensions, the boundaries of classification are distinguishable between the two most common classes. but much less obvious with the other two. This intuitively makes sense, since there are less points in the other two classes. However, it doesn't visually separate the clusters well in 2 dimensions. This supports the results found with eigenvalues: PCA does a better job separating phishing's components than with CE.

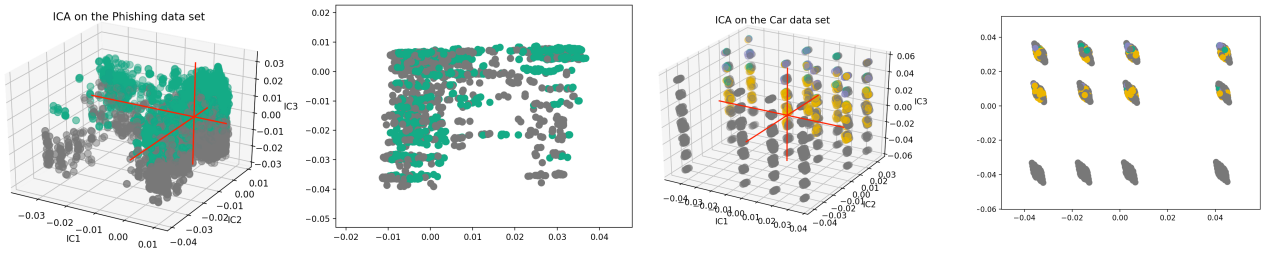
Independent Components Analysis

Similar to PCA, ICA is a feature transformation algorithm that tries to maximize independence rather than variance. It separates multivariate datasets into additive subcomponents with the assumption that all attributes are non-Gaussian signals and are statistically independent. In terms of runtime and accuracies, ICA performs similarly to PCA. Kurtosis is a measure of the combined sizes and probability within the two tails of a distribution. I plotted the kurtosis before and after running ICA on both datasets, as shown in the column





graphs below to evaluate performance further. Kurtosis for a normal distribution would be 3. We notice that there is a lot more variation in kurtoses for Phishing than there is with Car Evaluation. Interestingly, after ICA, Car Evaluation attributes come close to being Gaussian. Phishing attributes however, still indicate some mutual independence. This is likely in part caused by the larger number of attributes in Phishing. Not all of them would be related.

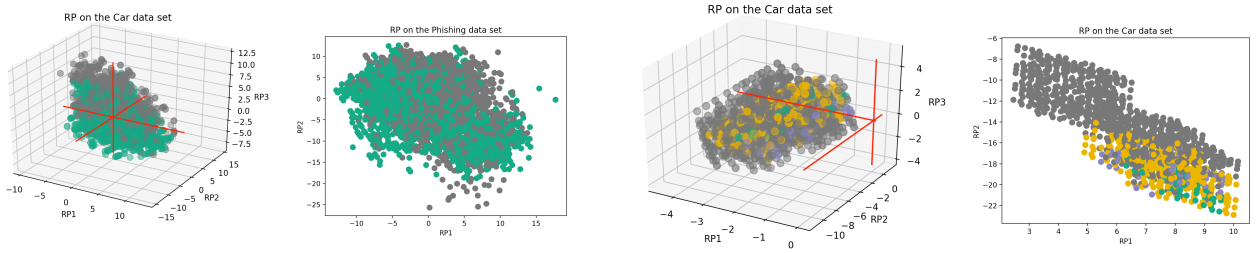


Above are 3D and 2D space visualizations of ICA clusterings. Overall we notice similar results as PCA in terms of successful separation of components. We still notice somewhat poor separation between Car Evaluation clusters. This intuitively relates to the kurtoses results, which demonstrated close to Gaussian attributes in the car dataset. ICA performs well on datasets with independent features, which CE doesn't seem to have as much of.

Randomized Projection

	Reconstruction Error
Phishing	0.58
Car Eval	0.46

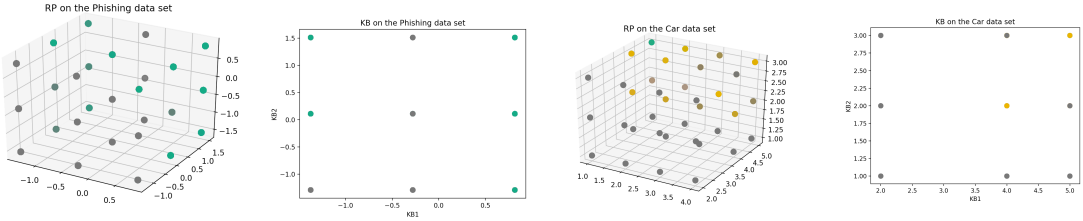
Random projection is a dimensionality reduction algorithm that generates a randomized matrix based on a target number of dimensions and uses it to apply a transformation to a smaller feature space. Doing this impacted performance, as we see in the increase in reconstruction error for both datasets in comparison to PCA. This was likely caused by a combination of features that weren't well correlated with one another. When this occurs, we notice odd cluster shapes



The graphs above are examples of 3D and 2D space visualizations of RP clusterings. We notice from the 2D projection that the Car dataset points are much better captured (components separated) in this case than with PCA or ICA. This appears to be extremely promising. Repeated runs of the algorithm however, didn't preserve this exact structure. Generally speaking, though, it still separated the dataset reasonably well on these low dimensions. Inspected the actual projection matrices (not captured) suggests projections that "best" separate classes heavily weight the four most important features found using the eigenvalues in PCA. Randomized projection on the Phishing dataset shows interesting results. We see somewhat well separated clusters in the 3D projection, but unfortunately, we see a great deal of variance in the random projections. The separating structure was mostly preserved, but there were some 3D projections in which distinct structure was not captured.

K-Best

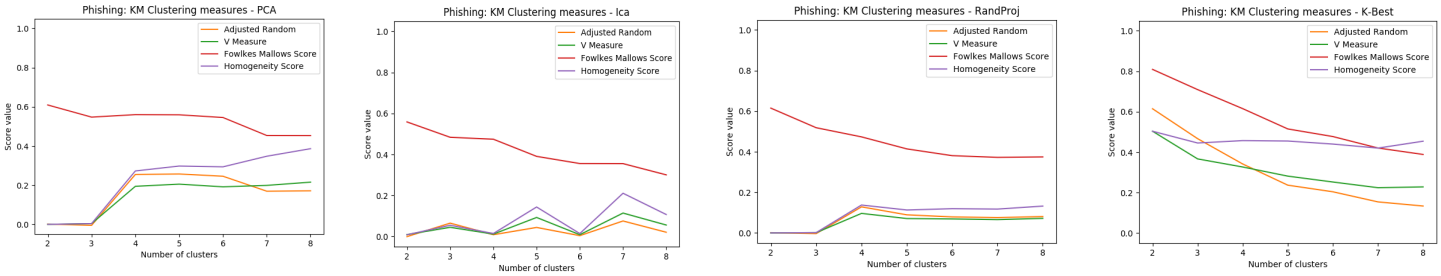
Select K-best takes as a parameter a score function, which must be applicable to a pair and returns an array of scores, one for each feature. It then simply retains the first k features with the highest scores. In my implementation, I selected the top 10 attributes which the highest ANOVA f-value.

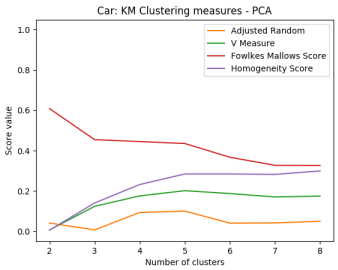


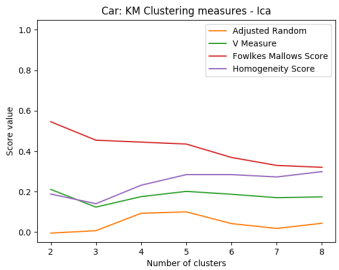
The clusters plotted on the left use the 3 and 2 highest scored features (for 3D and 2D projections respectively). Unlike the rest of the algorithms, we see much sparser results in lower dimensions using Select K-Best. We don't see well-defined clusters with either dataset either. This may just be a case of lower dimensionality, however. Still, this means Select K-best with low k values selects features that constrain the dataset to points that aren't very close to each other. Since "closeness" or distance has been a main determinant of clustering, Select K best allows for interesting comparisons.

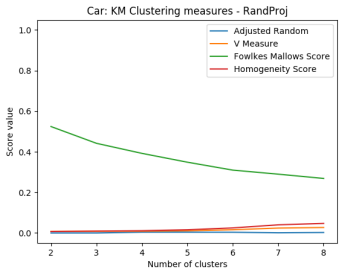
Clustering x Dimension Reduction

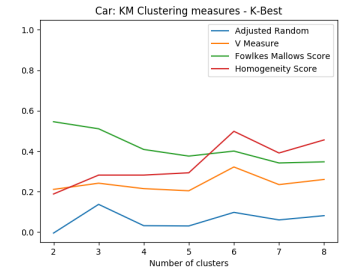
K-Means



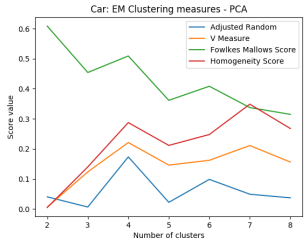


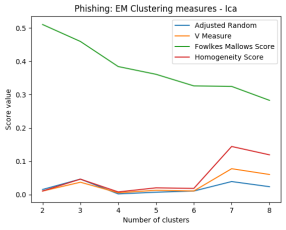


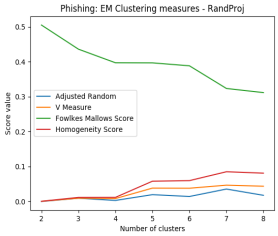


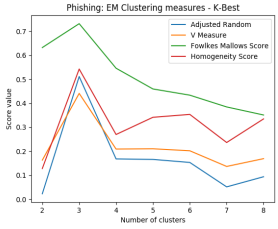


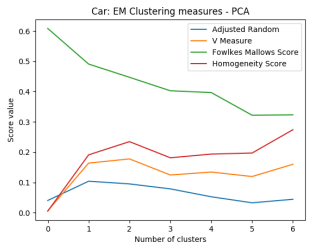
Expectation Maximization

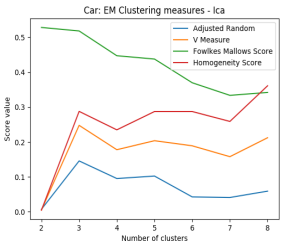


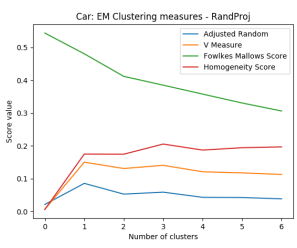


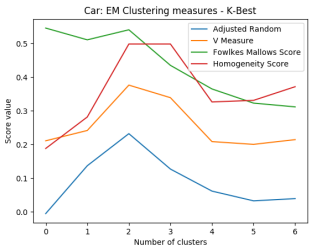






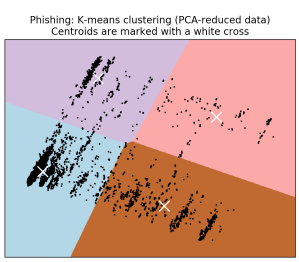


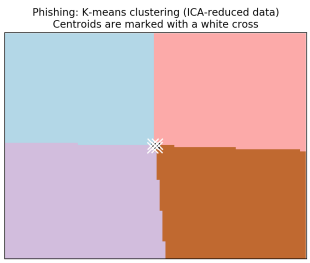


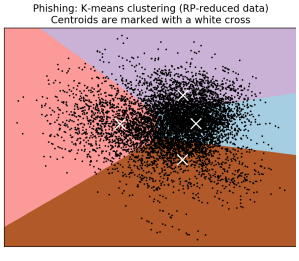


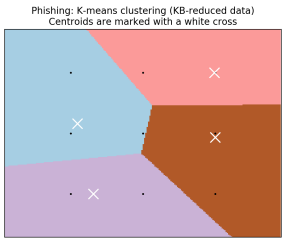
The above graphs represent similarity metrics for K-Means and EM on both datasets (top = Phishing; bottom = CE). We notice that, for Phishing with KM and EM, out of all the algorithms, ICA has the lowest adjusted random and V-measure scores - around 0.1 for both. KM and EM after PCA for Phishing results in significantly more similarity and precision/recall in comparison to just KM or EM (results presented in the first section). This is important because it occurs despite the high reconstruction error found with PCA on Phishing. We see this similarity particularly for 4 and 8 clusters. For the car dataset, none of the DR algorithms were able to reach the similarity scores of just KM or EM. ICA and PCA have nearly the same scores with the Car dataset, but ICA slightly outperforms PCA, which we expected from the clustering separation found in the previous section. Random Projection has the lowest similarity scores on both datasets. This is likely due to the un-likelihood of it capturing and combining the right features given that it attempts to do so randomly.

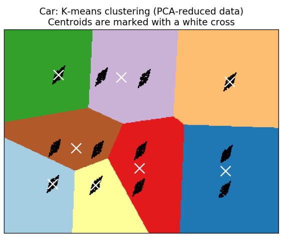
Clusters

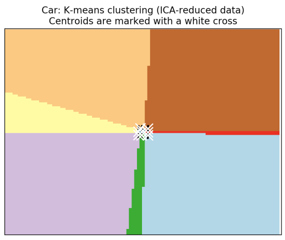


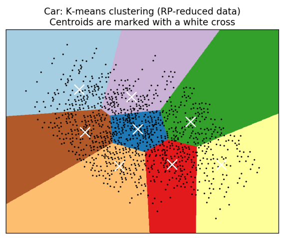


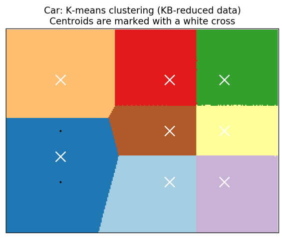








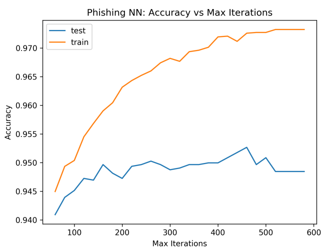


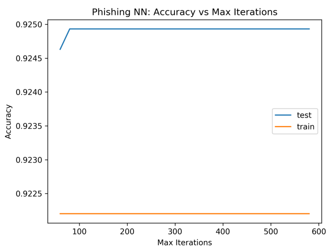


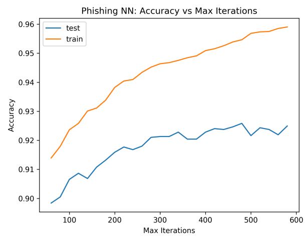
The graphs above visualize the effect of clustering on the data reduced to 2 components with each algorithm. We notice that after applying KM, we get different clusters than we did before with ICA. ICA with K means produced singular points when projected onto two dimensions, which doesn't provide us with meaningful information about the clusters themselves. We notice that RP led to similar results as it did before with no K-means clustering for both Phishing and Car Eval. So did PCA and Select K-Best.

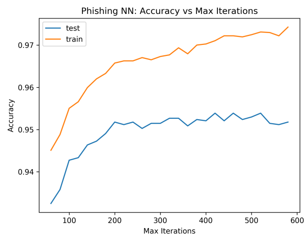
Similar results were obtained for EM, the graphs of which have been omitted for space.

Neural Networks









Num Components	PCA		ICA		Rand Proj		K-Best	
	Accuracy %	Time s	Accuracy %	Time s	Accuracy	Time	Accuracy	Time
4	86.1	5.39	85.2	0.84	79.2	5.22	92.5	1.68
8	93.8	5.5	92.2	1.35	80.2	5.74	93.7	2.63
12	94.0	5.99	92.25	1.24	90.6	5.76	94.3	4.64
16	94.9	6.15	92.3	1.24	91.8	6.01	94.9	5.79

Original	
Accuracy	95.7
Time	7.6

Num Components	PCA		ICA		Rand Proj		K-Best	
	Accuracy %	Time s	Accuracy %	Time s	Accuracy %	Time s	Accuracy %	Time s
KM	79.7	5.86	61.4	1.34	68.35	5.17	89.26	2.28
EM	59.9	6.23	63.25	1.72	56.6	5.74	57.3	2.73

In this final section I present the results of re-running my neural network from assignment 1 on the reduced datasets. I used a K value of 4 for K-Means. Altering this value produced next to no difference in results. We notice that accuracy decreases with the number of components, which is expected, since all the components result in complete data description, and the fewer the attributes used the greater the amount of information lost.

The obvious merit to reduction with no clustering however, is its relative efficiency. On a lot of my algorithms, I saw convergence of error rates very quickly, within 600 iterations, while my original neural network didn't see such convergence. In addition to iterations, we notice efficiency in terms of time. Select K-Best on half the features takes about 2 seconds less to train than the original data does but has almost the same test accuracy, for example. Since the curse of dimensionality makes it so that complexity and time/resources needed to train a model increase exponentially with more attributes, a decrease in 1% accuracy for 16 omitted attributes is excellent. This improvement is not just seen in Select K-Best. PCA demonstrates the same test accuracy, with one unit of time more than KB. ICA takes only 1.24 seconds to train and still reaches a test accuracy of 92.3.

Clustering information was added to the original data to test the effect of cluster analysis on NN performance; the results are summarized in the second table above. We can see that the classification accuracy of NNs trained with clustering is lower than NNs trained with just dimensionality reduction. The accuracies reached by EM are generally lower than KM accuracies, except in the case of ICA. The poor performance with EM clusters makes sense because the Fowlkes Mallows scores were lower for EM than K-Means for Phishing for all the algorithms besides ICA. Perhaps K-Means is not as suitable for ICA on the phishing dataset, since it produced the lowest K-Means accuracy. The time needed to train was a little longer in with clustering added as a feature than without, but still significantly less than the 7.6s training time of the NN with no clustering or Dimensionality Reduction. Accuracy on RP was the lowest of all algorithms with EM. We suspected that this would be the case in the previous section when comparing similarity metrics with Phishing, since Phishing doesn't demonstrate Gaussian properties. NNs applied to K-Best with K-Means perform most similarly to the original data. PCA K-Means results are not too far off either.

Conclusion

In this paper we've seen that dimensionality reduction algorithms play an important role in tackling the curse of dimensionality. They make classifiers like neural networks much quicker and easier to train, require fewer iterations and achieve adequate results. However, we also see that reducing dimensionality of data may make it prone to overfitting, and I some cases such as Randomized Propagation and Select K Best, it could also ignore important features for classification.

Based on my neural network results, I'd say that I don't recommend using clustering in the way this paper demonstrates on the phishing dataset, if the question is between using clustering and supervised learning methods. The increase in efficiency with training and testing time is far too small to make up for the decrease in accuracy. It does however demonstrate that there would be cases in which this improvement in time spent would justify using clustering even if it leads to slightly less accurate results.