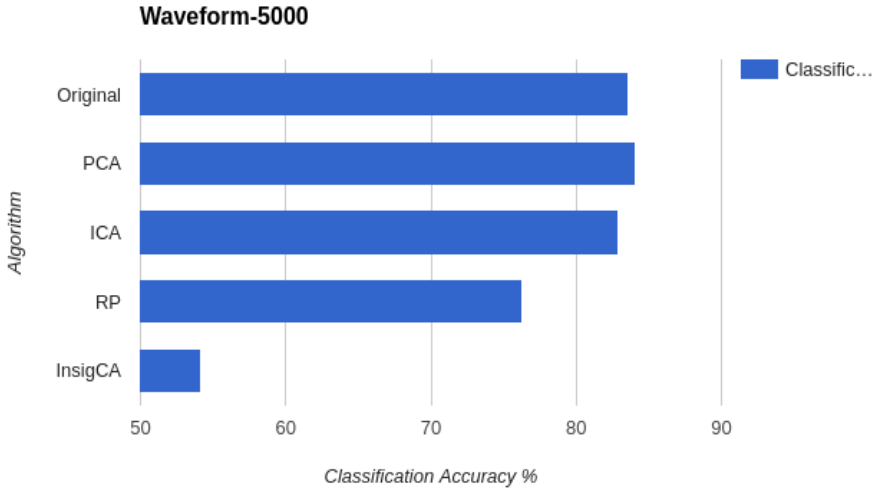
Robert Bradshaw
3/28/2017
CS 4641
Assignment 3: Unsupervised Learning

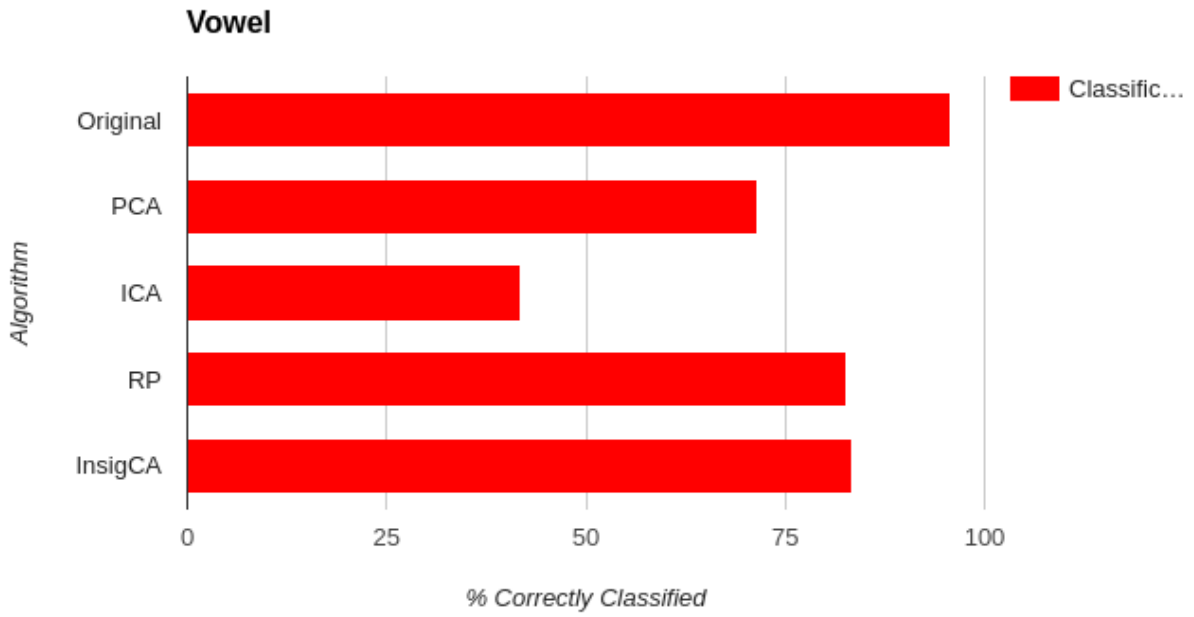
For Assignment 3, I'll be used the same two datasets I used for Assignment 1: Waveform-5000 (which deals with 3 classes of waves, 41 attributes with continuous values between 0 and 6, and 5000 instances) and the Vowel dataset (which deals with 11 classes, 14 attributes, and 990 instances). The vowel dataset is personally very interesting to me because it attempts to perform vowel recognition by taking in account sound factors (i.e. attributes) regarding the gender of the person speaking, the identity of the person speaking, and other attributes that are hidden from people using the dataset; I assume these are related to frequency, noise, the environment in which the sound files were recorded, and other audio-related attributes. The Waveform-5000 dataset is admittedly a very noisy and large dataset. Only SMO was able to perform well on this dataset in Assignment 1. I think this is a great dataset to use for this assignment because learning a bit about the nature of the dataset utilizing unsupervised learning techniques may help me when I try to reapply what I learned about the features towards tuning supervised learning parameters for the dataset in the future.

The dimensionality reduction algorithms I ran were PCA, ICA, RP (10 attributes kept), and Insignificant Component Analysis.

To start with, running WEKA's MultilayerPerceptron classification algorithm (i.e. Neural Network) on the Waveform-5000 dataset with 41 layers produced the lowest Root Mean Squared Error for me in Assignment 1. Running with the same configuration (10-fold cross validation) results in a classification accuracy of **83.56% (a Root Mean Squared Error of 0.315).** This will be our baseline of comparison for our dimensionality reduction algorithms.



Running WEKA's MultilayerPerceptron clasification algorithm (i.e. Neural Network) on the Vowel dataset with 28 layers produced the highest classification accuracy for me in Assignment 1. Running with the same configuration (10-fold cross validation) results in a classification accuracy of **95.7576% (and a Root Mean Squared Error of 0.0799).** This will be our baseline of comparison for our dimensionality reduction algorithms.



As a general note, classes are ignored for each clustering algorithm. Unsupervised learning is meant to help us learn about relationships between unlabeled data anyway – so I believe this is the best way to perform the analysis. The "variance covered parameter" for dimensionality reduction algorithms is standard: 0.95. Euclidian distance is employed as the distance metric – after reading up on the impacts different distance metrics have on the performance of algorithms, I learned that Euclidean distance does well when attributes have similar scales of measurement (as in "attributes with disproportionately larger scales of measurement can overwhelm attributes measured on a smaller scale"). Since the range of values for each attribute in the Vowel and the Waveform-5000 are consistent and are in a pretty narrow range, I deemed it most suitable to use Euclidean distance as the distance metric for our algorithms.

K Means

To determine the optimal K for K means clustering, I tried plotting the effect increasing K has on the Sum of Squared Errors (SSE) within the clusters produced for the original dataset and for the datasets produced via dimensionality reduction. I paid particular attention to the points at which the SSE began to diverge. I noticed that SSE was only capable of diverging with respect to K when looking at the transformed ICA datasets. I feel like this is an intuitive result due to the fact that ICA is keen on finding the

degree of independence between components (which translates well in my mind to the idea of clustering instances of data). This is shown in the following graph (the red values are critical points):

Waveform-5000 (K vs SSE)

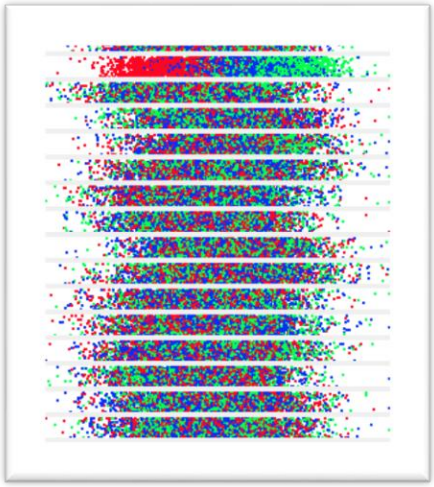
K	Original	ICA	PCA	RP (10)
2	3756.616394	3315.820166	3471.595216	798.1932171
3	3405.159102	**3225.099672**	3308.618203	681.5798155
4	3313.805448	3268.211022	3280.398274	649.350487
5	3247.129797	3168.131584	3251.449874	618.5064799
6	3200.97492	3141.023867	3223.33297	596.0582185

Vowel (K vs SSE)

K	Original	ICA	PCA	RP (10)
2	1629.76597	185.054764	117.6125439	289.5956098
3	1357.332522	127.1230221	99.88430935	240.4490161
4	1025.814093	65.09062824	86.81497861	210.5145407
8	808.9137597	39.31537283	51.57885107	157.126163
10	673.0896987	**26.13148896**	42.21959206	141.6544656
11	667.0922369	41.69815375	40.8900326	135.826807
12	663.8652581	41.46969145	39.54775595	128.8782159

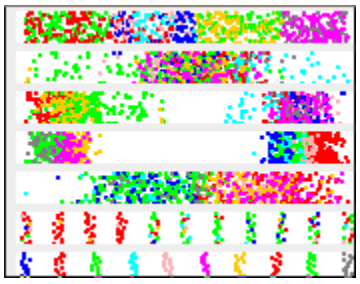
Also, you'll notice that for the both of the ICA transformed datasets that the divergence in SSE happens to occur on or very close to the total number of classes we have in each dataset (3 for Waveform-5000 and 11 for Vowel).

The WEKA SimpleKMeans clustering algorithm on ICA transformed Waveform-5000 returns a clustering distribution of 35%/32%/33%. The class distribution across the 5000 instances of this dataset is 33%/33%/33% (which got my hopes up because I was expecting much prettier gradients when I visualized the clusters). Waveform-5000 is an admittedly noisy dataset, so even at its most optimal clustering for SimpleKMeans, the visual representation of how the attributes are clustered relative to each other indicate that we still have much room for improvement:



ICA Transformed Waveform-5000 Cluster examples

SimpleKMeans on the ICA-transformed Vowel dataset performed very well here (having only 26.131 SSE at the most optimal K value). Look at how well the clusters are distributed across all of the reduced transformed attributes:



ICA-transformed Vowel (K Means)

Expectation Maximization

This algorithm works by essentially probabilistically determining the probability that an instance belongs to a cluster according the following formula:

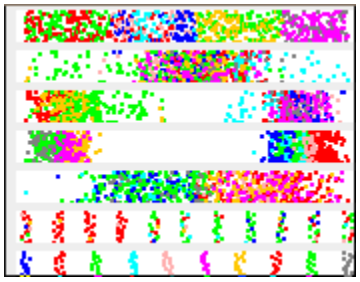
$$E[Z_{ij}] = \frac{P(x=x_i|\mu=\mu_j)}{\sum_{i=1}^{k} P(x=x_i|\mu=\mu_j)}$$

I utilized WEKA's default EM settings for this part of the assignment. Out of curiosity, instead of immediately using the optimal clusterings found by looking at the critical values in ICA transformed datasets (with regards to SSE), I let WEKA run on the default setting (where it attempts to use EM to find the optimal clusterings on its own using the training set). The optimal clustering that WEKA's EM algorithm found did not perform as well on the neural network classification as the optimal clusterings I found utilizing many iterations of K Means, analyzing the SSE, and applying my domain knowledge (of class distribution) to determine the correct number. I'll provide the results anyway:

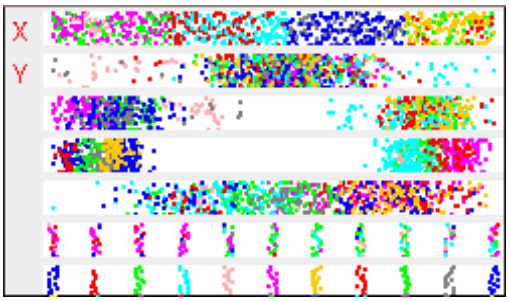
Waveform			
Ran On	**Number of Clusters Selected by Cross Val**	**Log Likelihood**	**Time(s)**
Original	11	-58.78086	2256.01
PCA	6	-52.03088	1055.13
ICA	6	-49.46945	1215.53
RP (10)	13	-32.61377	993.79
INSIG	2	-56.15786	82.18
Vowel			
Ran On	**Number of Clusters Selected by Cross Val**	**Log Likelihood**	**Time (s)**
Original	38	-7.54747	353.52
PCA	22	-5.77441	94.61
ICA	27	-0.91995	149.27
RP (10)	6	-16.57447	21.78
INSIG	28	-4.37937	254.5

As you can see, with Waveform, it doesn't come close to the number of classes. Also, the log likelihood (the value we are trying to maximize in EM) is so extremely negative on all the datasets that it's barely worth talking about. Look at the runtimes – even with all that time, WEKA's default EM settings couldn't sift through the noisiness of the Waveform-5000 dataset and provide meaningful clusters to help classify the data when we append the cluster information to the datasets later on in the paper.

I applied the EM algorithm to each dataset using the optimal clusterings found with K Means (3 for Waveform-5000 and 10 for Vowel). The following is the visualized cluster of the Vowel ICA-transformed dataset. Although this visual looks similar the K Means one above, there are distinct differences between all of the attributes (the clustering with respect to the fourth feature is a lot more distinct on the right side than it is in the K means clusters. This looks a lot more intuitive of a clustering than the fourth feature with K Means. Probabilistic clustering obviously takes longer but it does produce a better result in Vowel).



ICA-transformed Vowel (K Means)



ICA-transformed Vowel (EM)

Note: applying ICA to the datasets cause them to appear more Gaussian in nature (reduction in kurtosis).

Clustering Applied to Reduced Datasets' Neural Network Performance

The following tables are the results of running the neural network (multilayer perceptron in WEKA) algorithm on the datasets transformed via dimensionality reduction and including their clusters as features:

Waveform-5000 (K = 3)

Dataset	Root Mean Squared Error	Classification Accuracy
Original	0.3185	83.14
PCA	0.3184	83.32
ICA	0.3251	82.74
RP(10)	0.3306	76.22
InsigCA	0.5191	53.44

Waveform-5000 (EM = 3)

Dataset	Root Mean Squared Error	Classification Accuracy
Original	0.3191	83.18
PCA	0.3222	83
ICA	0.3251	82.74
RP(10)	0.3347	75.46
InsigCA	0.5048	53.24

Vowel (K = 10)

Algorithm	Root Mean Squared Error	Classification Accuracy
Original	0.1035	92.9293

PCA	0.1725	79.596
ICA	0.2654	43.7374
RP(10)	0.145	85.8586
InsigCA	0.0985	94.1414

Vowel (EM = 10)

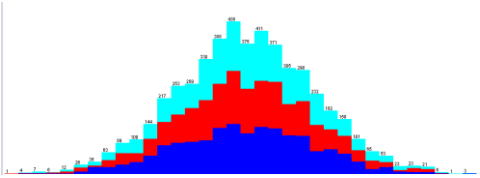
Algorithm	**Root Mean Squared Error**	**Classification Accuracy**
Original	0.0852	95.1515
PCA	0.203	71.5152
ICA	0.274	38.1818
RP(10)	0.1777	78.9899
InsigCA	0.1245	90.404

Some interesting things about the data above are that PCA-transformed Waveform-5000 dataset with respect to clustering assignments provides better neural network classification accuracy than the original dataset (with respect to clustering assignments) does with respect to K Means clustering. PCA doing better than the original for Waveform. ICA-transformed datasets consistently dramatically reduce the classification accuracy of the neural network. I think this because at an inherent level ICA is geared towards targeting the degree of independence between each attribute – PCA is meant to find relationships between features and also recognizes some features as more important than others. PCA's goal seems to fit more intuitively with the concept backpropagation as a means to predict outcomes based on the value and importance of certain attributes applied to specific instances – which may explain why it tends to do a lot better when applied to actual classification problems.

Another interesting thing is that on the K Means cluster-integrated Vowel datasets, Insignificant Component Analysis actually outperforms every other transformed dataset (and even the original!). I'm not particularly sure why this is.

ICA vs PCA

With the Waveform-5000 dataset, PCA actually results in a higher classification accuracy (lower root mean squared error) than every other dimensionality reduction algorithm. In addition to this, it actually results in **higher** classification accuracy than the original Waveform-5000 dataset when applied to WEKA's MultilayerPerceptron algorithm with 41 hidden layers (which was the optimal number of hidden layers found for the dataset in Assignment 1). PCA recognizes some components as more important than others using eigenvectors (as opposed to ICA -- which considers them all equally as important) whereas ICA emphasizes the "independence" of each attribute. ICA does well with non-Gaussian data. Since PCA outperforms it we can infer that the Waveform-5000 dataset can more easily be characterized as Gaussian than non-Gaussian. Visualizing the class distribution across most of the attributes of the original dataset in WEKA corroborates this presumption:



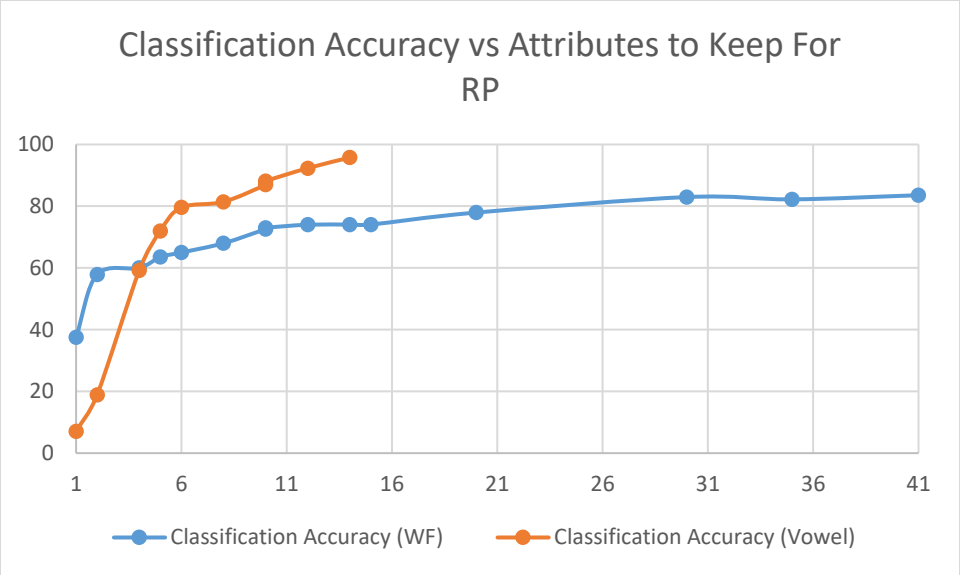
PCA utilizes eigenvalues to determine the “principal components” of a dataset. It attempts to find redundancies in attribute relations and summarize them in an entirely newly created attribute by analyzing the covariance of attributes relative to another to minimize error. The significance of each attribute is represented in the form of an eigenvalue corresponding to the eigenvectors found by diagonalizing covariance matrices produced during the iterations. The eigenvalue of largest magnitude is our “principal component”.

The eigenvalues of Waveform-5000’s PCA decomposition indicate a heavy skew. As in, the principal component has a corresponding eigenvalue of 7.94866, the penultimate principal component has a corresponding eigenvalue of 3.2321, and the rest of the eigenvalues for the generalized attributes have corresponding eigenvalues of 1 or less.

The eigenvalues of Vowel’s PCA decomposition indicate less skew than the eigenvalues of Waveform-5000. The order of eigenvalues from the principal component onwards looks like: 3.44445, 2.57046, 2.28923, 2.07221, … It makes sense that PCA consistently performed worse than the original datasets when applied to the neural network classification problem – it was unable to find disproportionately meaningful generalizations of attribute relations like Waveform-5000 was (which explains why in Waveform-5000, PCA increases classification accuracy when applied to the neural network).

Randomized Projections

I was curious about how the number of attributes to keep affected classification accuracy of the neural network, so I ran RP with a bunch of different values (0 < n < total number of attributes). I used the RPRunner file from the github code I included.



Using only 1 attribute results in pretty low accuracy with RP. However, RP logarithmically increases its potency with each attribute you decide to keep. Keeping 1 attribute for RP on the Vowel dataset results in a classification accuracy of approximately 7% -- but keeping 5 attributes results in a classification accuracy of approximately 72%!

Some other interesting things:

- Running the WEKA MultilayerPerceptron algorithm on the reduced datasets always resulted in a lower run time than when it is run on the original datasets. The curse of dimensionality states that analyzing data in high dimensional spaces becomes exponentially hard of a problem and less and less feasible as the dimensions grow. It makes sense that run time decreased for our classification algorithm when we reduced the number of attributes (i.e dimensions) to analyze.
- When applying RP to Waveform-5000 (and its transformed variants), it would usually translate the overall graph representing class distributions to right a little, and skew the values to the right.

References

- WEKA
- http://www.colorado.edu/engineering/CAS/courses.d/ASEN6519.d/Lectures.d/Lecture10_11.6519.pdf
- http://www2.cs.uregina.ca/~dbd/cs831/notes/clustering/clustering.html
- http://stanford.edu/~cpiech/cs221/handouts/kmeans.html

- http://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues
- https://github.com/theJenix/ml-project-3