Assignment 3: Unsupervised Learning and Dimensionality Reduction

Shaikh Shamid

April 5, 2015

1 Introduction

In unsupervised Learning we learn from data without prior knowledge of its classification. Dimensionality reduction simply means reducing the features/attributes of the given dataset without losing information. We learn from data, also we improve the data by dimensionality reduction technique. In my previous work[1, 2], I used two different datasets, one with binary classification and another with multiclass classification. I will continue using these dataset for this project. In this case however, our data does not have classification labels. For clustering, I will be using k-means clustering and expectation maximization (EM) in order to classify the data. For dimensionality reduction, I will be using principal component analysis (PCA), independent component analysis (ICA), Random Projections (RP), and Principal Factor Analysis (PFA). In the end I will compare the results of one of the datasets on a supervised learning neural network algorithm, and explain the merits and demerits that the clustering and dimensionality reduction algorithms may have on these data sets.

2 Algorithms

In this section I will briefly explain all the algorithms considered in this project, and what publicly available software was used in order to implement them.

2.1 Clustering Algorithms

For the clustering algorithms, I will use the implementations in the publicly available R library[3].

2.1.1 K-Means Clustering

In K-Means algorithm we choose k points from the data and assign those to be the centroid of each cluster. Then, classify each data point to one of the clusters based on a distance metric (such as Euclidean and manhattan distance). The next step is to recalculate the average of each of the clusters and assign that to be the new cluster centroids. We reclassify the data points, and continue to iterate until it converges. In order to implement it I used the R flexclust package [4], which has the advantage of tuning distance metric.

2.1.2 Expectation Maximization (EM)

Expectation Maximization (EM) is a probabilistic model. As it's name suggest, the algorithm alternates between two phases: Expectation and Maximization. During the E-step, the algorithm chooses a model or parameterized function that lower bounds log likelihood probability everywhere. During the M-step, it moves to a new parameter set that maximizes the function. I used the R mclust package [5] for implementing this algorithm.

2.2 Dimensionality Reduction Algorithms

Dimensionality Reduction algorithms projects the input data into a lower dimensional space prior to classifying it with learning algorithms.

2.2.1 Principal Component Analysis (PCA)

Principal Component Analysis finds a new eigen basis of the system. In this eigen space the distance between the clusters of information is maximized with reduced number of orthogonal components which is actually a linear combination original data attributes. I used the implementation available in stats package of the core R library[3].

2.2.2 Independent Component Analysis (ICA)

Independent Component Analysis is a higher-order method that linearly projects the input data by increasing the separation between each of the components from one another, and the they are almost statistically independent. Estimation of the model consists of two steps: specifying the objective function (also called the contrast, the loss function, the cost function), and the algorithm to optimize the objective function. The implementation of ICA is available in the R e1071 package[6].

2.2.3 Random Projections (RP)

Random Projection projects a dataset with n attributes to a k-dimensional space, where k is inherently lower than n. Given that its projection of the data to lower dimensions, this method works well and guaranteed to improve the computation time of the algorithm (although this doesn't say much for its accuracy). Depending on the projection, the algorithm may not be heavily affected in its accuracy, but reasonably, accuracy will tend to decrease as n approaches 1. I used the implementation available in python scikit-learn's random projection module[7].

2.2.4 Principal Axis Factor Analysis (PFA)

Like PCA, principal factor analysis (PFA) is also a linear method which assumes that the measured variables depend on some unknown, and often unmeasurable, latent common factors. The task here is to uncover such relations, and thus can be used to reduce the dimension of datasets following the factor model. The implementation of PFA is available in the R psych package[8].

3 Data acquisition

Both data sets come from the UCI Machine Learning repository [9]. I did some preprocessing of both data sets before applying clustering and dimensional reduction algorithms.

3.1 Abalone Data Set

The abalone data set has 4177 instances and 8 different attributes. The preprocessing step here is to define three abalone age groups: 0-8 rings as infant, 9-11 as adult and greater than 11 rings as old.

3.1.1 Adult Data Set

The adult data set has over 22000 instances and 14 features of mostly categorical variables. I did some preprocessing of the original data by discarding couple of repeated attributes and clean the categorical values by giving good names. In order to apply clustering and dimensional reduction algorithms the most important preprocessing step was to factorize the nominal data to numerics.

4 Implementation: Clustering Algorithms

In the unsupervised clustering algorithms the important task is to find the number of clusters to extract from the input data. Although both datasets have those class labels, but I removed those classification attributes from the data in order to properly evaluate the cluster sizes for each of clustering algorithms. After successfully obtaining the optimal cluster sizes I tell clustering algorithms how many clusters to extract from the data points. And then I took the advantage of comparing it with the known classification attributes present in the datasets.

4.1 K-Means Clustering

For k-means clustering, I find the optimal number of clusters by calculating the within cluster sum of squares (wss) for different cluster sizes. The plot is given in Figure 1a, which shows that as cluster size increases, within cluster sum of squares decreases as it should be. But if we look closely we see that upto cluster size three the within cluster sum of squares decreases linearly, and after that it decreases more or less exponentially. So the elbow at cluster size three tells that the optimal number of cluster should be three for the abalone dataset, which also justifies my preprocessing steps of abalone dataset where I classified the abalone data labels into three different age groups described in abalone dataset section. The next point I investigated for k-mean algorithms is to choose the proper distance metric. Figure 1b shows that the euclidean distance metric is giving the highest accuracy estimates. So, for the rest of the k-means clustering analysis I stick to euclidean distance measure. The accuracy I get for abalone dataset is 60%, and for adult dataset is 71%. The important observation here is that the accuracy depends on the choice of the initial centroid of the k-mean clusters. This make sense because for some bad centroid choice data points may well be misclassified. In order to improve the accuracy I had to loop over the random number generators.

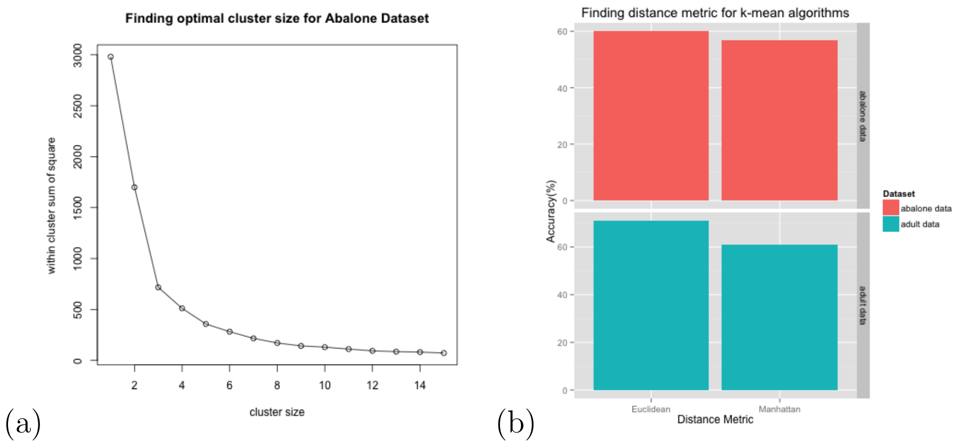


Figure 1: (a) Plot of within cluster sum of square vs cluster size for abalone data. (b) Manhattan and Euclidean distance metric comparison for both dataset based on accuracy estimates. For both dataset Euclidean outperformed the manhattan distance measure.

4.2 Expectation Maximization (EM)

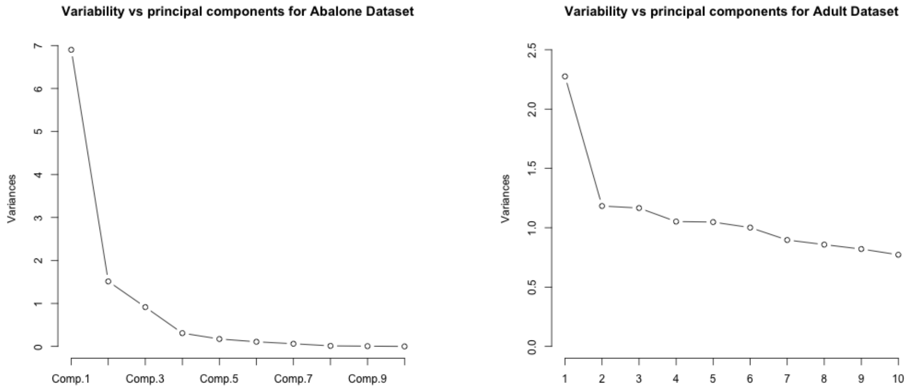
In expectation maximization, I expected to see a better performance than with k-means clustering. But, the maximum accuracy I get for both the datasets is around 46 %. Due to the fact that the abalone dataset is small in size, the accuracy suffers greatly for this dataset even after telling the right number of cluster sizes. But the adult dataset is relatively large still accuracy is low. I believe this is because of the converted nominal attributes to factors variables for this dataset. There is some issue with clustering algorithms for nominal attributes. EM was also run without a cluster specified, and returned very poor results for both dataset. Performance is also an issue for this algorithm. I hope the dimensionality reduction algorithms may help the performance, and that is what I will be implementing next.

5 Implementation: Dimensionality Reduction

The Dimensionality Reduction algorithms focus on changing the structure of the input data prior to processing it with a learning algorithm, which in our case, will be processed with a multilayer perceptron (neural network).

5.1 PCA

In PCA the first step I choose is to find the optimal number of components for both the datasets. For this, I plot the variances vs number of principal components, which is given in Figure 1. By elbow analysis I conclude that the optimal number of components for abalone data is 4, which describes the 96% variability of the data points. So, PCA reduces the abalone dataset from 10 attributes to 4 attributes. The elbow analysis for adult dataset is little bit subtle. From the plot it looks like the elbow is at component size 2, but it only describes 30% variability of the data points. For this dataset I end up taking first 7 components which describes 72 % variability of the data.

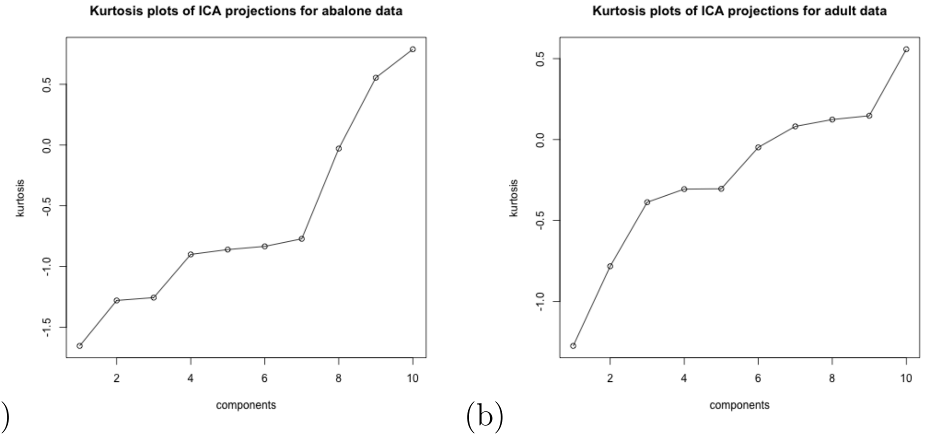


(a) (b)

Figure 2: (a) Plot of variances vs number of components to find the optimal number of principal components for abalone dataset. (b) Plot of variances vs number of components to find the optimal number of principal components for adult dataset.

5.2 ICA

For ICA projection I analyzed its output by applying negative kurtosis function. I ordered the ICA output according to their kurtosis values. I keep the components which have negative kurtosis values meaning that their histograms are non-gaussian. According to kurtosis analysis as shown in Figure 3, ICA reduces the abalone dataset from 10 attributes to 7 attributes, whereas for adult dataset ICA reduces 12 attributes to 5 attributes.

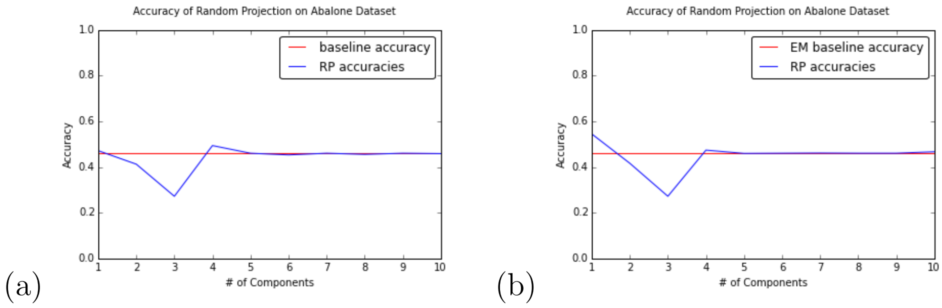


(a) (b)

Figure 3: (a) Plot of kurtosis vs number of components to find the optimal number of independent components for abalone dataset. (b) Plot of kurtosis vs number of components to find the optimal number of independent components for adult dataset. In this plot out of 12 components I plot first 10 components to see their negative kurtosis values accurately.

5.3 Random Projections

For Random Projections, I calculate the accuracies for different RP components and compare them to baseline accuracy (no dimensionality reduction applied) when applied to k-means clustering algorithm. Figure 4 shows that, for both dataset RP accuracies are approximately equal to the baseline accuracy at component 5. So, for this dimensionality reduction algorithm the number of dimension reduces to 5 for both the dataset.

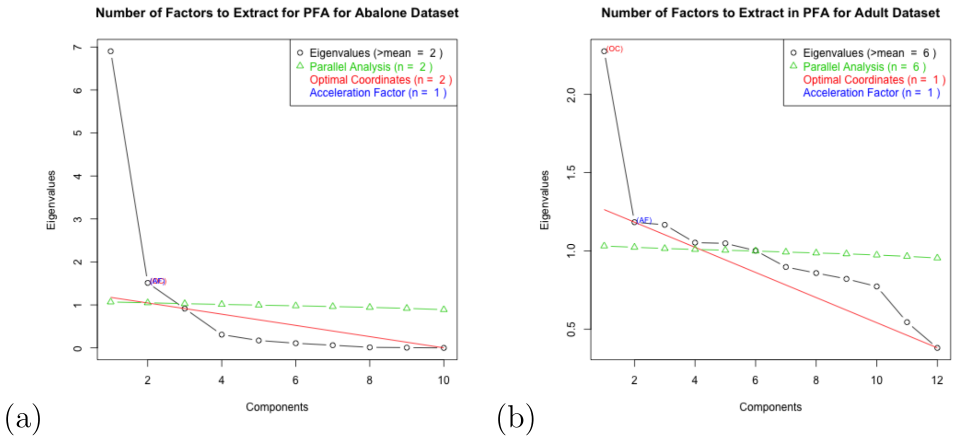


(a) (b)

Figure 4: (a) Plot of accuracies vs number of components and comparison to k-mean baseline(no dimensional reduction) accuracy. (b) Plot of accuracies vs number of components and comparison to EM baseline accuracy for abalone data.

5.4 Principal Factor Analysis

Lastly, for Principal Factor Analysis, I calculate the eigenvalues for different factors for both dataset as shown in figure 5, which also shows the optimal number of factors to extract from. For abalone the optimal number of factors are 2. I choose the optimal number of factor to be also 2 for adult dataset.



(a) (b)

Figure 5: (a) Plot of eigenvalues vs number of factors to find the optimal number of factors to extract for abalone data (b) Plot of eigenvalues vs number of factors to find the optimal number of factors to extract for adult data.

6 Dimensionality Reduction and Clustering

In this section I will discuss the effects of clustering algorithms when applied to the projecting data coming from after applying dimensionality reduction algorithms. Reproducing these results after dimensionality reduction produced different levels of accuracy and evaluation time, from which we can conclude that they produce different clusters. However, the performance difference wasn't so different for me to interpret that the clusters were greatly different before dimensionality reduction. I think the clusters are more compact due to the reduction of dimensions and some data points may change their clusters, and therefore reflect a change in the final results. PCA is the most negatively affected by the change in dimensionality reduction in terms of both performance time and accuracy, and I think this because a loss of components that may have negative impact on the accuracy and performance time. Surprisingly, as we can see from table 1 and 2 , PFA does the best in terms of accuracy time trade-off.

Dim. Red Algorithm	Red. Dim	Accuracy	Time (s)
PCA	4	54.4%	0.39
ICA	7	51.6%	0.33
RP	5	50.4%	0.33
PFA	2	55.6%	0.30
None	10	60.0%	0.33

Table 1: Comparison of Algorithm Performance for abalone data when applying k-means algorithm to the projected data. None means no dimensionality reduction applied.

I summarize the effects of k-means clustering algorithms when applied to the projecting data after applying dimensionality reduction algorithms in 1 for abalone dataset and in 2 for adult dataset. For both datasets the accuracy drops. The performance time is comparable for small-sized abalone dataset, but for relatively large adult dataset, performance of the algorithm improves greatly for almost all the cases, as it should be. For expectation maximization algorithm, I run the same experiment and find that the outcome is similar to what we see for k-means. It is important to note that increasing the number of attributes also increases the amount of time the algorithm takes to compute. The best dimensionality reduction algorithm I find is PFA. In PFA the algorithms chooses smaller number of principal factors, and for this reason the clustering algorithm is much faster than others, as

Dim. Red Algorithm	Red. Dim	Accuracy	Time (s)
PCA	7	64%	2.48
ICA	5	58%	2.31
RP	5	56%	2.37
PFA	2	60%	2.17
None	12	71%	2.39

Table 2: Comparison of Algorithm Performance for adult data when applying k-means algorithm to the projected data. None means no dimensionality reduction applied.

Dim. Red Algorithm	Red. Dim+cluster	Accuracy	Time (s)
PCA	8	55.1%	0.31
ICA	6	80.23%	0.33
RP	6	100%	0.34
PFA	3	85.84%	0.30
None	13	82.76%	0.33

Table 3: Comparison of Algorithm Performance for abalone data when applying k-means algorithm treating cluster as a new feature. None means no dimensionality reduction was applied to the data.

evidence from table 1 and 2. Also for this dimensionality reduction algorithm the accuracy drops is reasonable.

In order to see effect of performance time and accuracy when cluster information already present, but with the projected data, I run the k-means clustering algorithm after adding the cluster to data as a new feature. I summarize the result in table 3, which shows a great improvement in the accuracy estimates. Again PFA dimensionality reduction algorithm outperformed all the other algorithms in terms of accuracy and computing time. Overall, the performance time is comparable when cluster information is not present in the data. But the accuracy is incredibly high, even better than baseline accuracy. By baseline I mean there are no dimensionality reduction algorithm performed to the data.

7 Neural Network Classifier

In order to evaluate all of the clustering and dimensionality reduction algorithms, I ran a neural network Classifier on all of the results. The first interesting phenomenon that the neural network test demonstrates is the computing time vs accuracy trade-off. As can be demonstrated in table 4, for all of the dimensionality reduction algorithms it leads to better accuracy, and hence classification, but the computing time suffers due to the complexity of neural network classifier, which weighs the parameters adequately to obtain the best accuracy estimates.

Dim. Red Algorithm	Red. Dim	Accuracy	Time (s)
PCA	7	67.8%	5.92
ICA	5	72.8%	6.02
RP	5	49.4%	5.37
PFA	2	65.7%	4.90
None	12	73.4%	6.21

Table 4: Comparison of Algorithm Performance for abalone data when applying neural network Classifier to the projected data. None means no dimensionality reduction was applied to the data.

In order to see effect of applying neural network classifier on the performance time and accuracy when cluster information already present, but with the projected data, I run the classifier after adding the cluster to data as a new feature. The result is summarized in table 5, which shows a perfect classification for all the dimensionality reduced projected data. The performance time is also improved in this case.

Dim. Red Algorithm	Red. Dim+cluster	Accuracy	Time (s)
PCA	8	100%	5.51
ICA	6	100%	5.36
RP	6	100%	5.02
PFA	3	100%	4.61
None	13	100%	6.02

Table 5: Comparison of Algorithm Performance when applied neural network classifier to the projected data considering cluster as a new feature. None means no dimensionality reduction was applied to the data.

8 Conclusion

In conclusion, we get the decreasing in processing time of clustering algorithms when dimensionality reduction is applied. In the specific case of our datasets, it does not seem to be the case that accuracy was improved, but there were cases in which performance remained approximately the same. We learn that the unsupervised learning methods work well for data sets that have an underlining clustering distribution, but when we compare and contrast models, the differences in their performance can be partly attributable not to their differing structure, but to the different levels of tuning effort we invest in them. Also the types of algorithms that work best are problem-specific.

9 Acknowledgements

I would like to recognize the work by [9] for the UCI Machine Learning Dataset Repository which contained both of the datasets discussed in this paper. I would also like to acknowledge the R programming library[3] which provides almost all my implementation of the dimensionality reduction and clustering algorithms.

References

[1] https://github.com/sshamid/ML_projects

[2] https://github.com/sshamid/ML_projects2

[3] R Core Team (2014). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/

[4] Friedrich Leisch. A Toolbox for K-Centroids Cluster Analysis. Computational Statistics and Data Analysis, 51 (2), 526-544, 2006.

[5] Chris Fraley, Adrian E. Raftery, T. Brendan Murphy, and Luca Scrucca (2012) mclust Version 4 for R: Normal Mixture Modeling for Model-Based Clustering, Classification, and Density Estimation Technical Report No. 597, Department of Statistics, University of Washington.

[6] David Meyer, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel and Friedrich Leisch (2014). e1071: Misc Functions of the Department of Statistics (e1071), TU Wien. R package version 1.6-4. http://CRAN.R-project.org/package=e1071

[7] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[8] Revelle, W. (2015) psych: Procedures for Personality and Psychological Research, Northwestern University, Evanston, Illinois, USA, http://CRAN.R-project.org/package=psych Version = 1.5.1.

[9] FRANK, A., AND ASUNCION, A., UCI machine learning repository (2010).