Nimisha Devanagondi, Anjali Ramesh, Harmeena Sandhu, Ashna Sood, Urmi Suresh
Professor Marcelo Mattar
COGS 118B: Introduction to Machine Learning II
6 June 2021

A Comparison of Unsupervised Algorithms on a Bean Clustering Task

Introduction and Motivation:

In this study, we aimed to classify different kinds of beans using a comparison of Unsupervised Machine Learning algorithms—specifically K-Means Clustering and Principal Component Analysis, using the Dry Bean Dataset. This is an important study because using these unsupervised classification methods allows unlabeled data to become labeled and known to humans in a much more efficient and effective way than hand classifying would be. The "best" algorithm depends on the data set itself, and finding the most accurate and efficient algorithm prevents a myriad of misclassifications, while also collecting labels at a relatively fast rate. Additionally, over 40,000 bean varieties exist around the world, with some being edible and others being poisonous [1]. When cultivating a variety of beans, it is important to know if the bean type falls under the category of "toxic". Furthermore, different cultures around the world use different bean varieties in their cuisine, and knowing which variety thrives in the said region particular to said cuisine keeps authenticity within cultural meals, and maintains the natural order of the food chain in those areas [2].

The Dry Bean Dataset is the dataset that is explored in this project [3]. This dataset consists of 17 attributes with 13611 samples. The attributes of the data are the following: Area(A) describes the area of a bean zone and the number of pixels within the boundaries of the picture, Perimeter (P) describes the bean circumference, Major Axis Length (L) is the largest length of the bean, Minor Axis Length (I) indicates the largest length bean perpendicular to the main axis, Aspect Ratio (K) indicates the relationship between L and I, Eccentricity (Ec) describes how irregular the bean's eclipse shape is, Convex area (C) shows the number of pixels in the smallest convex polygon that can contain the seed, Equivalent Diameter (Ed) is the length of the diameter of the circle around the bean seed area, Extent (Ex) is the ratio of the pixels of bounding box to area of the bean, Solidity (S) is the ratio of pixels in the convex shell of the bean, Roundness (R) shows how close to circular / spherical the bean truly is, Compactness (CO) measures rounds of the object, Shape Factor1 (SF1), Shape Factor 2 (SF2), Shape Factor 3 (SF3), Shape Factor 4 (SF4), and Class (Seker, Barbunya, Bombay, Cali, Dermosan, Horoz, and Sira) labels the type of bean. While performing our analysis, PCA was used to reduce the dimensionality of the dataset, as well as the K-Means algorithm to perform classification, which would ultimately cluster the data by bean type, given the 16 features (excluding "class"). We then will report the differences of results between our hand coded algorithm for K-Means, and that of the Sklearn package [4].

Related Work:

A recent paper in *Intelligent Data Engineering and Automated Learning,* titled "Cluster Analysis of High-Dimensional Data: A Case Study", addresses a similar problem to the one that was explored in this project. Namely, having a dataset with high dimensionality, and how to accurately and efficiently analyze this data using unsupervised machine learning. The authors of this paper took a similar approach to this experiment, which was finding the most important dimensions using principal component analysis (PCA) and then running a K-Means clustering algorithm to group the data into corresponding clusters [5]. Although the process used in this experiment was specifically for the Dry Beans dataset, this paper shows that the combination of these algorithms could be useful in generalizing to other high dimensional datasets. This paper verifies and supports our methods, showing that our problem, while not unique, definitely has an accurate and efficient solution.

The PCA algorithm can be broken down into the following steps: computing the mean of the data, computing the covariance matrix and then normalizing its eigenvectors, sorting those eigenvectors and eigenvalues in the order of decreasing eigenvalues, computing principal components, reconstructing the zero-meaned pattern, and finally adding back the mean [6]. Essentially, PCA is reducing the space or dimensionality of the data being worked with.

The K-Means algorithm consists of the following steps: one starts with the initial guess (or knowledge) of *k* number of clusters, then points will move closer to the mean nearest in proximity to it, and thus the centers of the clusters will update. K-Means is complete once the centers of each cluster stop updating [7] . It is essentially a clustering algorithm that partitions data into *k* number of clusters.

Methods:

The full process consisted of cleaning up the dataset, performing exploratory data analysis (EDA), and carrying out an analysis of the dataset.

1. Data Cleanup: The dataset chosen was the "Dry Bean Dataset" found in the UCI Machine Learning Repository. The data contains 13611 data points and 17 attributes including the class of each bean. Without the class, there are 16 attributes that were explained earlier in the paper. Some columns were renamed for uniformity, and rows with missing values were dropped, as well as the 'Class' column. This left a dataset without null values and with only the 16 features.

2. EDA: The first step was to print out the general descriptive statistics of the dataset. This includes values that summarize the central tendency, dispersion, and shape of the bean dataset's distribution (*Figure 1*). Correlations between each attribute were also printed and placed into a heatmap to clearly visualize which variables were most highly correlated with each other (*Figure 2*). This aids in determining which and how many attributes are essential when performing PCA, and reducing the number of principal components used.

3. Data Analysis: The steps taken for analysis were to write and implement the K-Means algorithm from scratch, compare that with Sklearn's K-Means algorithm, write and implement the PCA algorithm from scratch to re-perform K-Means on the dimensionally-reduced dataset, compare that with Sklearn's PCA algorithm, and finally compare the results of the various K-Means algorithms to see if the reduction in dimensions would better cluster the bean types.

K-Means from Scratch: Each helper method represents a different step of the K-Means clustering process. The calcSqDistances function returns an array that contains the distances between each data point and each of the corresponding existing cluster centers. The determineRnk function returns an array with a one-hot encoding (of 0s and 1s) for each datapoint depending on which data point belongs to which cluster. It does this by looping through the squared distances array returned by calcSqDistances and for each row, setting the column with the smallest distance equal to 1. The recalcMus function returns an array containing the new cluster centers. It finds these new cluster centers by looping through the number of clusters and for each data value updating the cluster centers based on where the 1 is in the one hot encoding. After this, the means of the new clusters were taken and this resulted in the new cluster center array. Finally, the runKMeans function performs K-Means with various k clusters. It starts by initializing cluster centers by a method of randomly choosing points out from the data. Then it calls calcSqDistances, determineRnk, and recalcMus in a loop. Finally, the old means and the new recalculated means are compared and if they are significantly different the loop begins again but if they are unchanged the K-Means algorithm ends. Using a function provided in Cogs 118B Homework 2, the first and last K-Means iteration were plotted to visualize the clustering process (*Figure 3* displays 7 clusters). In terms of the EM algorithm, the expectation step would be finding the cluster memberships through determineRnk, and the maximization step would be updating the cluster centers based on the newly updated cluster memberships which is done in recalcMus. Essentially, the EM algorithm helps with maximizing the likelihood of data.

K-Means Clustering using Sklearn: This dataset was also clustered using the scikit-learn library in order to compare it to the accuracy of K-Means algorithm written from scratch. When running this, the elbow method was utilized to verify that k = 7 clusters was the best number of *k*s to cluster this data with, as there are 7 bean types. To do this, inertia was calculated, which is the sum of squared distances of samples to their closest cluster center, and these values were plotted on a graph to determine the optimal number of clusters.

PCA From Scratch: In the PCA algorithm the basic breakdown of principal component analysis steps was followed. After running the PCA algorithm on the dataset, the top principal components of the dataset were found, and stored into a new dataframe. The original K-Means algorithm was then run on this new dataframe using the top 5, 10, and 16 components (*Figures 4, 5, 6* respectively). Using the dimensionally-reduced data with the top 10 principal components, a closer K-Means clustering analysis using various k values from 2-10 was performed. By doing

this, a more streamlined version of the dataset can be seen and how it was clustered with these new parameters.

PCA using Sklearn: The dimensionality of the dataset was also reduced using the built-in PCA function from Sklearn in an attempt to compare the results to the PCA algorithm written from scratch.

Results:

Using k = 7 clusters, we looked at the distributions of the predicted clusters for the K-Means algorithm written from scratch and the K-Means algorithm ran on the dimensionally-reduced data with 10 principal components. From this we observed that both algorithms clustered the data in a similar way in that both could not completely isolate one bean type per cluster. This was able to be verified as the original dataset included the bean type for every datapoint, but in order to perform these unsupervised methods, the labels were omitted. It is important to note that each time the K-Means algorithm is ran, a different subset of beans are assigned to a different numeric value from 0-6, so a certain number cannot always be associated with a bean type. Additionally, without the actual predicted class labels from the K-Means algorithm, we could not perform various error metrics such as Mean-Squared Error, Sum-of-Squared Error, or Root Mean-Squared Error as we had initially planned on. Because of this, we were not able to directly compare the distributions of each cluster between both K-Means runs, but after comparing across averages, we were able to see that within the created clusters after the PCA algorithm was ran, the dominant class made up a slightly higher percentage of the distribution of the cluster versus those in the original K-Means algorithm. For example, within cluster 3 in the original algorithm, the dominant class made up 37% of the total distribution, while after PCA, it made up 81% of the total distribution (*Figure 7*). Thus, it is evident that reducing the dimensionality of the data aided in the clustering of the various bean types. If further time was permitted, we could have done a deeper analysis and breakdown of the bean classes with using the top 5 principal components to reduce the dimensionality of the data to see if that even better separated out the bean types.

Because we held the prior knowledge that there were 7 bean classes in the dataset, we knew that 7 clusters would be the ultimate number to separate out the data. However, we additionally took the approach of not considering this fact and allowed the K-Means clustering visualizations and elbow method to determine the optimal number of clusters before the data becomes overfitted. This is carried out in two manners. Firstly, within the K-means clustering algorithm written from scratch before PCA was applied, we visually plotted the created clusters and analyzed how each additional cluster from 2-10 aided in separating out the data. It was evident from the visualizations plotted for k = 5 that the data was ideally split up (*Figure 8*). Secondly, as seen in the graph (*Figure 9),* when using the elbow method with the Sklearn K-Means algorithm, we determined that 4 *k*s was the optimal number for this dataset. Although this was not in line with our original hypothesis that the data would be clustered into 7 clusters,

one for each given Bean type, this did make sense with our analysis of the distribution of each predicted cluster. Since many of the bean classes had features that were very similar to each other, 7 classes might have overfitted the data even when there were 7 distinct types of beans. Because of this, it makes sense that this K-Means algorithm's optimal number of clusters was around 4-5.

Discussion:

Through our project, we learned that using PCA to reduce the dimensionality of the data and then running the K-Means algorithm produces slightly higher, but generally similar results to when K-Means is run on the dataset without doing dimensionality reduction. This is what we expected to happen, as the point of PCA is to be able to reconstruct a somewhat similar dataset using only the top principle components, which will reduce the dimensionality of the data. When working with high dimensional data, such as we did, this would be extremely helpful to get accurate and efficient results that can be analyzed while still preserving computing power and time. However, in the future something we could further explore would be to use even fewer principle components in our reconstruction and breakdown the actual vs. predicted number of beans in each class to see if the K-Means results are even better. An interesting idea would be to see what the minimum number of principal components, or dimensions, needs to be in order to cluster the dataset at least with the same, if not better, accuracy as the original K-Means algorithm with no dimensionality reduction.

An extension that we would have liked to make to this project would be to also use a multivariate Gaussian mixture model to group the data into clusters, and compare this algorithm to both our original K-Means algorithm and the K-Means algorithm with reconstructed data. By doing this, we could have compared three different methods of clustering and grouping our data, and ranked the accuracy of each algorithm or combination of algorithms. This would help us to use the theoretical information that we learned in class in a practical setting, and be able to visualize the results of each algorithm on the Dry Beans dataset.

Another improvement we could have made to our project would be to streamline our K-Means and PCA algorithms, which were written from scratch, to be able to run faster and more efficiently. As of now, these algorithms were written with each step clearly detailed and multiple functions, but there are definitely ways to improve these algorithms to run more efficiently and with the same accuracy as the scikit-learn library's built in functions. Streamlining our algorithms would help us to use the code on even higher dimensional data, which typically takes greater amounts of processing time.

Contributions:

Each member of the group contributed equally to the code, paper, and presentation.

Code:

https://github.com/ashnasood/Comparison-of-Unsupervised-ML-Algorithms-on-Clustering-Task

References

[1] Sandridge, Suzanna. “Toxic Beans.” *StateFoodSafety Resources*, July 2014, www.statefoodsafety.com/Resources/Resources/toxic-beans.

[2] “Beans Around the World.” *Bean Institute*, beaninstitute.com/beans-around-the-world/.

[3] Koklu M., Ozkan Ilker A. UCI Machine Learning Repository [https://archive.ics.uci.edu/ml/datasets/Dry+Bean+Dataset]. Irvine, CA: University of California, School of Information and Computer Science.

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

[5] Bean R., McLachlan G. (2005) Cluster Analysis of High-Dimensional Data: A Case Study. In: Gallagher M., Hogan J.P., Maire F. (eds) Intelligent Data Engineering and Automated Learning - IDEAL 2005. IDEAL 2005. Lecture Notes in Computer Science, vol 3578. Springer, Berlin, Heidelberg.

[6] Jaadi, Zakaria. “A Step-by-Step Explanation of Principal Component Analysis (PCA).” *Built In*, 1 Apr. 2021, builtin.com/data-science/step-step-explanation-principal-component-analysis.

[7] Yildirim, Soner. “K-Means Clustering - Explained.” *Medium*, Towards Data Science, 3 Mar. 2020, towardsdatascience.com/k-means-clustering-explained-4528df86a120.

Appendix

Figure 1:

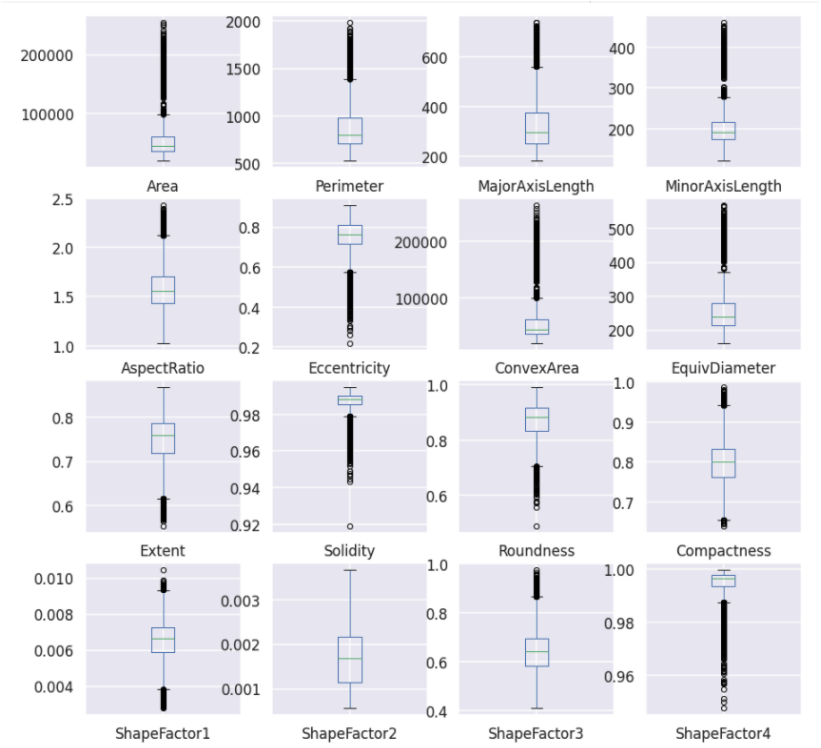

Figure 2:

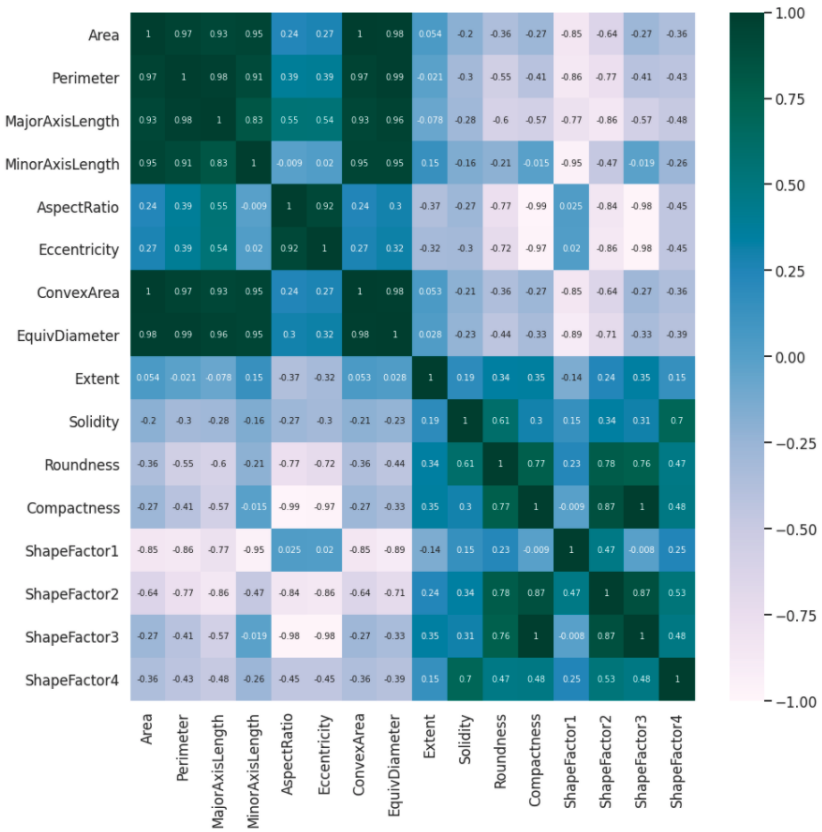

Figure 3:

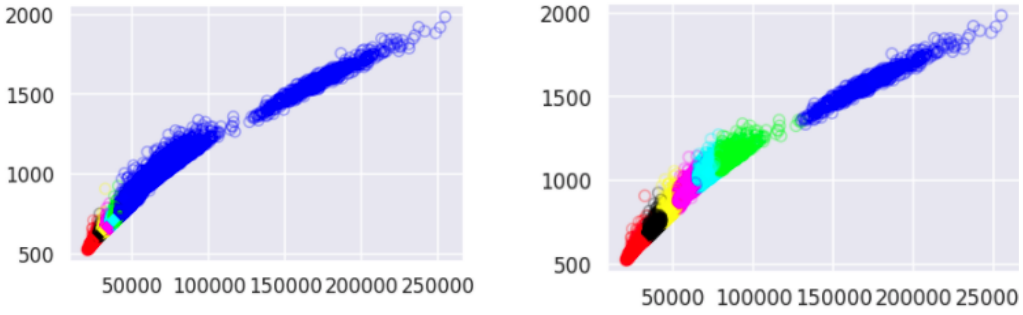

Figure 4:

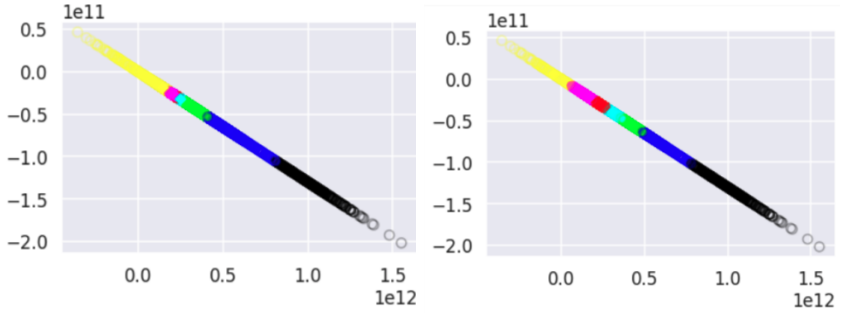

Figure 5:

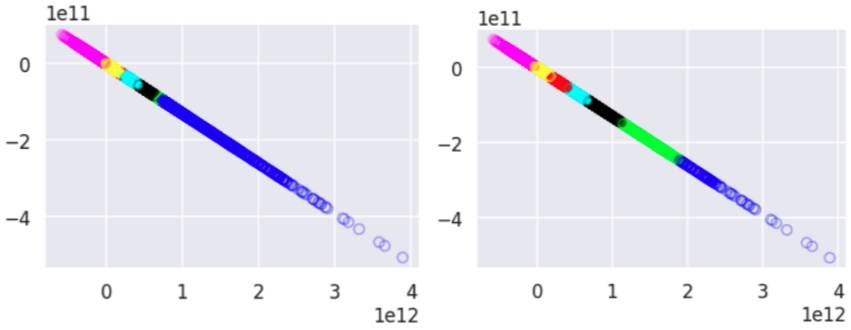

Figure 6:

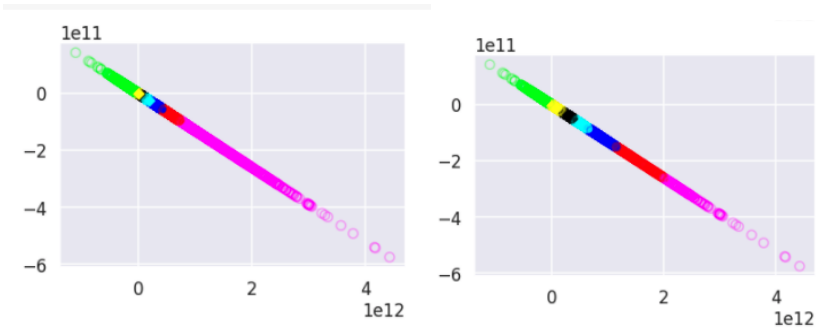

Figure 7:

```
Number of beans in Cluster 3: 1.0    3680
Name: 3, dtype: int64
SEKER        1375
DERMASON     1213
SIRA          945
HOROZ         144
BARBUNYA        3
Name: Class, dtype: int64
```

Vs.

```
Number of beans in Cluster 3: 1.0    1455
Name: 3, dtype: int64
CALI         1181
SEKER         108
BARBUNYA       87
HOROZ          63
SIRA           15
BOMBAY          1
Name: Class, dtype: int64
```

Figure 8:

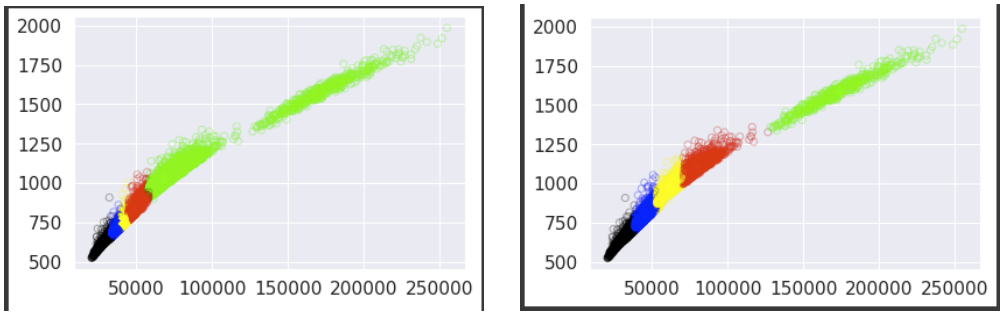

Figure 9: