PCA and K-Means

Muhammad Banatwalla

November 13, 2020

1 Introduction

For this homework, we were tasked with exploring clustering using k-means. We will attempt to address the font classification problem with the solution of k-means clustering. After finding our clusters using this algorithm, we will evaluate its success.

1.1 Description of Data

We used the same data set as we did in HW3. The 3 fonts we are working with are Century, Ebrima, and Gill, coming from the ”University of California Irvine Repository for Machine Learning Datasets.” For Class 1 we used the Century font with a size of 1649, for Class 2 we used the Ebrima font with a size of 1723, and for Class 3 we used the Gill font with a size of 1459. Class 1, Class 2, and Class 3 were unioned into a larger data set we call DATA with size 4831. Further, we then standardized DATA to get a new set called SDATA. Each case in the dataset describes numerically a digitized image of some specific character typed in the particular font.The image associated to this row has 400 features (or numerical descriptors). These 400 features represent the gray levels of the 400 pixels of the specific image. The last column is the fonts column that contains the labels of the fonts specifying what font each case belongs to.

2 K-Means

K Means Clustering splits the data into k groups that are similar to each other. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster’s centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum.Thus, the notion of similarity is derived by how close a data point is to the centroid of the cluster.Here we will attempt to use our fonts data set with the gray levels of the 400 pixels of specific images of the font and apply k-means to try to identify the three different fonts. Here we will attempt to use our fonts data set with the gray levels of the 400 pixels of this specific image of the font and apply k-means to try to identify the three different fonts. Here are the performance levels for the different values of k:

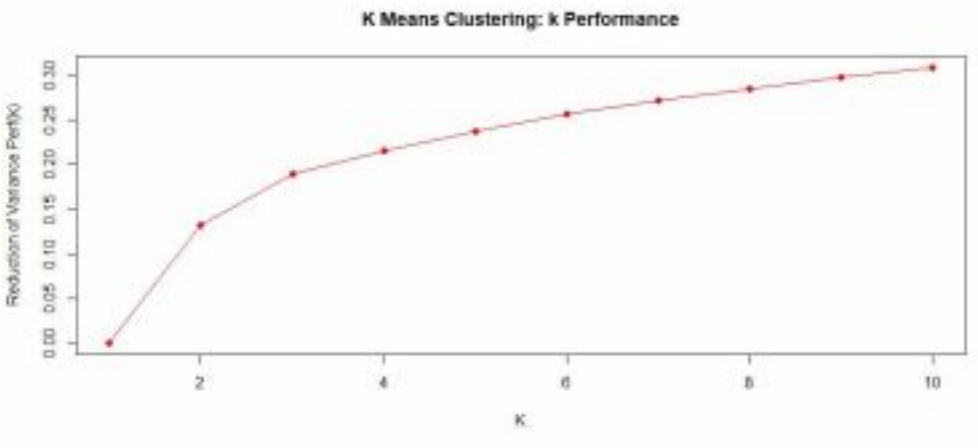

Figure 1: K Means Clustering Performance Curve

K Means Clustering is an unsupervised machine learning algorithm, it will not have reference to the known font labels. In order to determine the optimal number of clusters for our data we used the elbow method. The “elbow” is the point where the sum of squared errors between the data points and their centers begins to flatten out as k increases. We ran k means clustering for k values from 1 to 10 without the fonts labels. We divided the within sum of squares and the dispersion for the whole data set for each run to get the Q(m) relative dispersion. corresponding quality:

k	1	2	3	4	5	6	7	8	9	10
Q(m)	1	.87	.81	.78	.76	.74	.73	.71	.70	.69

Table 1: Cluster Quality for k-values

We also graphed these values to implement the elbow rule:

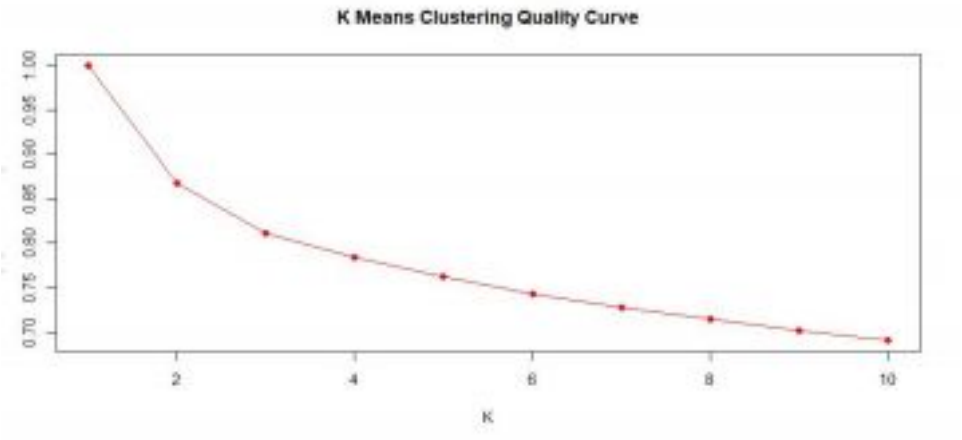

Figure 2: K Means Clustering Quality Curve

The ideal k value is when both k and the relative dispersion are small. From the graph we determined 3 to be the best number of clusters for our dataset. There is a huge decrease in relative dispersion with k=3 but after that the relative dispersion does not go down as quickly.

2.1 PCA and Visualization

We would like to visualize our clustered data. The problem is that our data contains 400 features. It would be difficult to decide which features to plot against each other. Therefore, we used our PCA results from HW3 and combined these results with the k-means algorithm. We used the first three principal components in order to visualize our clustering results. Since, a lot of the information from our dataset is held in the first three principal components. We take a look at the coordinates for the centers of each of the 3 clusters for the first 3 principal components:

	PC1	PC2	PC3
1	-2.7	6.8	.84
2	-4.2	-3.9	-.24
3	14.3	-1.2	-.72

Table 2: Cluster Centers for 3 Principal Components

To calculate the center coordinates for our three principal components we took the scalar product of the coordinates of cluster centers of the original 400 features (matrix 3x400) and the first three eigenvectors (400x3) from our PCA results. We then visualize these centers:

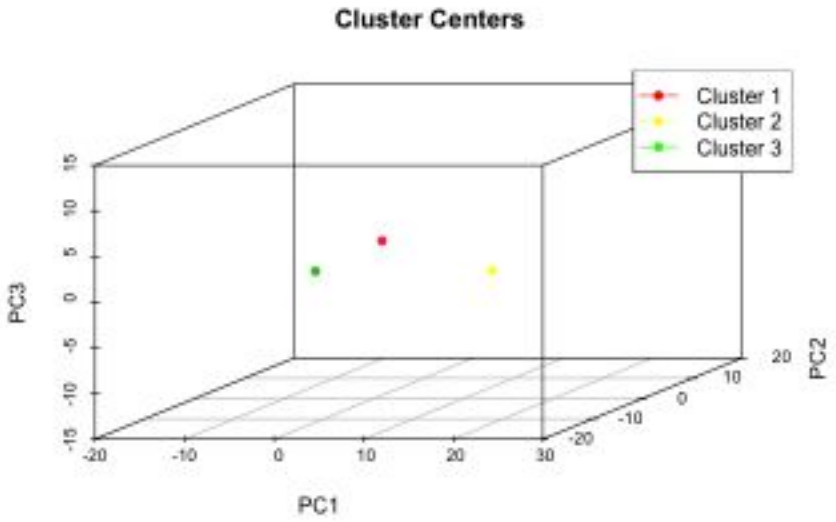

Figure 3: Cluster Centers

The points represent our three centeroids, they give us a rough idea about where our clusters will be. We can see that they are reasonably distanced and help us get an estimation of the shape of our data when taking into account the three top PCA's.

In dimension reduction by principal component analysis, principal component one ex plains the largest amount of variation in our data (17.5%), principal component 2 explains the second most variation (9.9%), and principal component 3 explains the third most varia tion in our data (5.4%). Therefore, a lot of the information from our dataset can be found in the first three principal components. We plotted below each data point in principal component 1, 2, and 3 colored according to its cluster assignment:

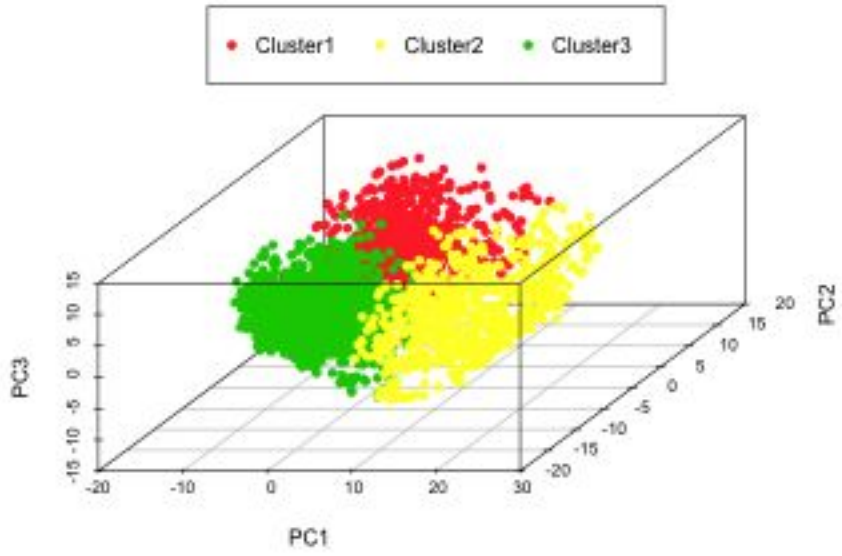

Figure 4: K Means Clustering

Visualizing the plot above it seems that the k means algorithm grouped data points that are close to each other in the same cluster.

Cluster 1 had size 2303, cluster 2 had size 1028 and cluster 3 had size 1500. With cluster 1 being the largest, we plotted the first three principal components from cluster 1 by color according to their class:

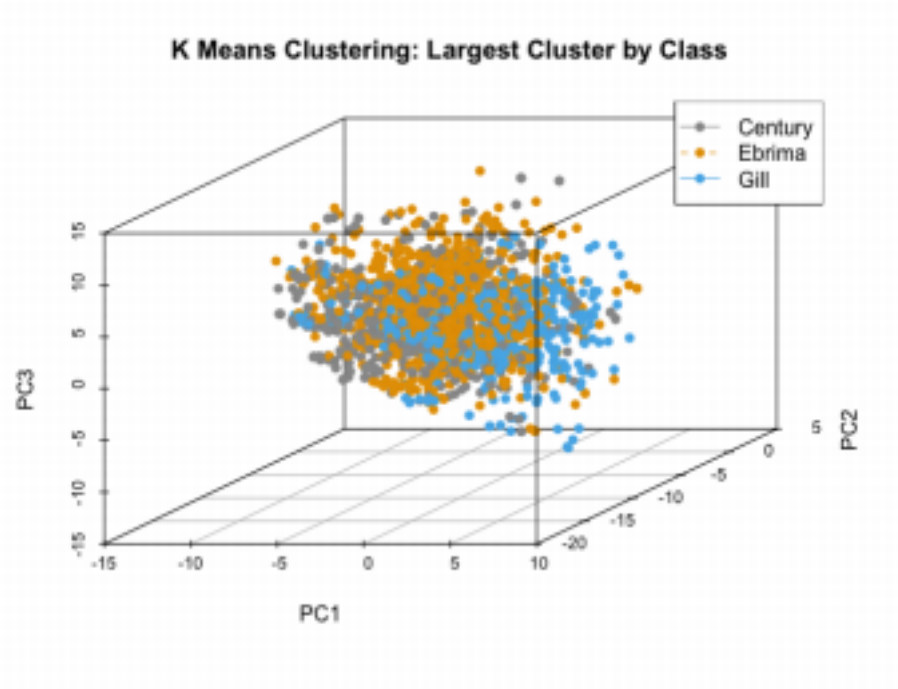

Figure 5: Largest Cluster by Class

By looking at the plot it seems that cluster one doesn't appear to have a clear winner- a font that appears the most. This could cause problems as we will likely have clusters with high impurity that do not discriminate our fonts well

2.2 Cluster Quality

Through different methods, we evaluate the quality of these clusters to see how well they reveal the underlying true classes of the data. The gini indices are a way to measure this cluster quality.

Cluster	1	2	3	All 3
Gini Index	.6209	.3508	.6314	56.68

Table 3: Gini Indices

This shows that the probability a font has been misclassified in our cluster one is 0.6209 or 62.09% This is very high! This shows that the probability a font has been misclassified in our cluster two is 0.3508 or 35.08%. For an unsupervised learning task this is rather good. Cluster 2 was our most accurate cluster. Cluster three was the most impure for us. The

probability that this cluster misclassified a font is 0.6314 or 63.14%. The total impurity for our cluster was 0.5668 or 56.68%. This means for any cases we can be 56.68% sure that it would be classified in the wrong cluster. This is rather high and does not give us confidence in the performance of our k-means algorithm!

We also investigate the frequency of each class within the clusters. Below is the frequency table:

Font	Cluster 1	Cluster 2	Cluster 3
Century	39%	12%	42%
Ebrima	45%	9%	39%
Gill	16%	79%	18%

Table 4: Cluster Frequency

Each case was assigned to a cluster. The table displays how frequently each font was assigned to each cluster. For cluster one ebrima was assigned more frequently then the rest, for cluster two gill was assigned the most, and for cluster three century was the maximum. Even though ebrima and century were the maximum for cluster one and three respectively, it wasn't by much. It seems that century and ebrima were almost evenly spread out through cluster one and three. Nevertheless, we will use the clustering to define predictors using the most frequent class in each cluster as the prediction for that cluster. Since cluster one had maximum ebrima characters assigned to it it will predict ebrima, since cluster two had maximum gill characters it will predict gill, and so on. Then we will create a confusion matrix below to examine our results.

3 K-Means and Prediction

Using the results from clustering, we will attempt to use these clusters as predictors for the fonts. Using the most frequent font in each cluster, we evaluate its performance as a classifier and look at its confusion matrix:

	Predicted: CENTURY	Predicted: EBRIMA	Predicted: GILL
Actual: CENTURY	38%	54%	8%
Actual: EBRIMA	35%	60%	5%
Actual: GILL	19%	26%	56%

Table 5: Confusion Matrix Using K-Means Prediction

In order to check how accurate our unsupervised clustering was in matching the character with its particular font we built a confusion matrix. The table compares the true class font for the cases versus the predicted class from our k means clustering. The overall accuracy for the k means clustering was 51% which is bad. The classification of the font century was the worst with only 38% classified currently. 54% true class century were classified by k means to be font Ebrima and 35% of true class Ebrima were misclassified as century. The interesting thing to note is that even though the k means clustering confused many century

and ebrima fonts with each other it didn't misclassify many of them as the gill font only 5% and 8%. However, many of the characters that belong to gill font were misclassified as century and ebrima 19% and 26%.

It is important to note the difference between the frequency table and the confusion matrix. The frequency table merely states the frequency to which each font appears in each cluster. Whereas the confusion matrix assigns each cluster as a predictor for a specific font and uses the truth labels of the fonts to see how accurately these clusters work as predictors.

4 Conclusion and Further Suggestions

Our results were not good from our K-Means algorithm. Only 51% of our cases were classified correctly. Since, k means clustering picks the clusters by minimizing the dispersion but doesn't try to control the impurity of the clusters we attempted to find clusters where both the impurities of the clusters and the dispersion are minimized. If the impurities of the clusters are minimized this could help the classification of the k means clustering. We ran k means clustering with k=3 one hundred different times to get one hundred different clusters. Unfortunately, due to the time constraints and computational times it takes we weren't able to determine the impurities of the clusters in time. However, in the future we could attempt to find the impurities of the clusters so that our k-means clustering algorithm can also take into consideration the impurity of the clusters. Another thing to note is that K-means clustering doesn't have access to any labels like previous classification methods we have used so it essentially "blindly" tries to predict groups. For k- means to be effective in the classification task, it relies on cases of the same class being close to each other. Unfortunately that is not the case for our data and thus k-means fails to be as effective. Our classification task would be served better by using tree based algorithms such as random forest or decision trees or other more advanced algorithms such as neural networks, support vector machines or deep learning.