Sara Bawale
Professor Kathryn Leonard
COMP 347
Final Project
13 December 2019

Shape Hierarchy:
Using K-Means Clustering to Find and Classify User Types

Sara Bawale, Caroline Dahl, Giuliana Zanutta

I. Introduction

One of the most whimsical pastimes is cloud-watching, where we can find dragons, castles, even spaceships among the clouds. But how do we recognize these clouds as distinct shapes? To explore this question, we used an open access database for 2D shape structure investigation compiled by Carlier, Leonard, Hahmann, Morin, and Collins. The database contains 1,255 different types of shapes with 2,861 different users annotating the shapes. Not every user annotated every shape, so there is variability in the types of annotations. Each annotation is a string of digits from 0 to 3, and each number represents one of the four colors users utilized to represent the shape hierarchy. Each shape had at least 24 annotations.

One thing we noticed about this data set was that there were some users who, instead of creating a reasonable shape hierarchy, just annotated the shapes in the most bright and eye-catching way possible. Their annotations were sporadic instead of thoughtful, and we didn't want their "artistic" creations to remain in the dataset. Our issue is that we want to remove problem data from the data set, but without going through all 42,000 annotations by hand. We decided that using unsupervised learning was the best chance we had at identifying artists from the dataset and separating them from non-artists. In this project we used K-means Clustering to see if an unsupervised learning model can identify artists as a cluster, and perhaps create further distinctions in the dataset we were unaware of.

II. Background

Preprocessing Data:

The first part of this project was to find features for our data that could potentially group our data into distinct clusters. We created a script that goes through each data point and creates features for each percentage of color the shape contains for each annotation.

Clustering Model:

In order to build our models we are trying to see if increasing the amount of clusters causes the matlab model kmeans to create an artist cluster. The strength of this method is that it is easy to comprehend what we are asking the model to do. We can also display the shapes the model clustered together to see if they visually match what we are looking for in artists. Unfortunately, flaws in our model conception are much more apparent. While our method is easy to comprehend, at the same time we have very little control over what types of clusters the models will make. Prior to building the models we don't know if our features will be enough to have the model correctly cluster an artist group from the dataset, or if it will create clusters completely unrelated to our goal. However, we still believe using an unsupervised model such as this is best for this data set because of the fact that all of our data is unlabeled, and we want to create classification groups for this large data set. Using clustering will produce possible groups for similar labels.

Clusters We Want:

We are looking specifically for distinct groupings of annotations, and hoping that one grouping will be classifiable as an artist grouping. Similar annotations being grouped together would look something like this:

Artist cluster *All black cluster* *Pink 4 Lyfe cluster*

Fun Fact: these shapes were actually grouped together in an early rendition of the model.

III. Methods

Preprocessing Data:

We started by importing the entire shapes hierarchy dataset as a table. For each annotation, we calculated the percentage of annotations that were labeled each color and created a new column for each color. The four possible colors were:

black: 0	**pink: 1**	**green: 2**	**red: 3**
main component	secondary component	tertiary component	detail

Because we wanted to try to find as many defining features of artists as possible, we tried to capture the eclectic nature of their annotations. The best way we were able to replicate this was by creating another feature that counted the amount of times an annotation changed color. To further illustrate, when encountering something like '0112333', the amount of changes is 4 because the individual digits changed four times. Although this feature is imperfect — the annotations are not in a sequential order — we found that capturing some semblance of color changes was an important feature. In order to have the color changes work for every shape, we changed the feature slightly from amount of changes, to the percent of changes in one annotation as different shapes had annotations of different lengths.

K-Means Clustering:

In order to create the model for our classification problem we used matlab's kmeans model. It returns an index with the cluster it classifies each data point to.

Parameter Choices:

In order to create the best clustering model, and in particular a clustering model that would cluster artists together, we decided to create multiple models using kmeans. K stands for the number of clusters in the model, and we increased the amount from k = 2 to k = 10. For each set of clusters we selected a subset of the annotations in that cluster to see if there were any discernible themes. We also adjusted the 'Replicates' parameter in matlab's kmeans model to try different random starting points for the centers of the clusters.

IV. Results

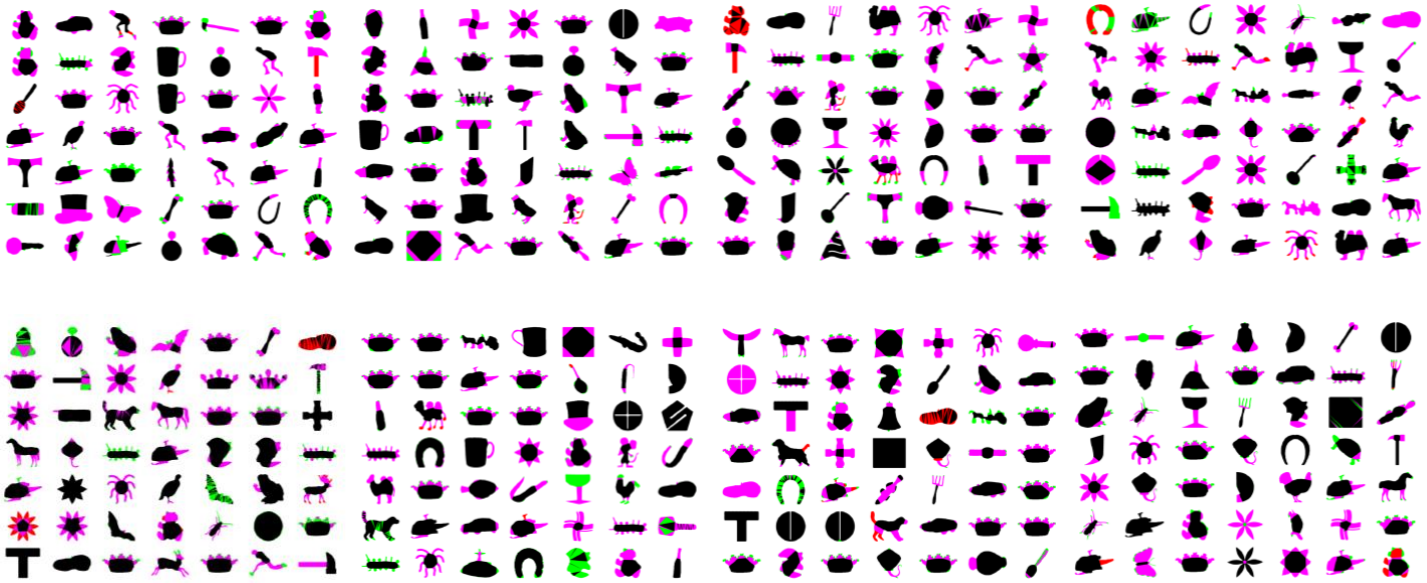
K = 2:5

We found that the k values 2, 3, 4 and 5 did not produce significantly meaningful groupings, or an artist grouping. Artists were dispersed throughout several clusters, and clusters were mostly distinguished based on the dominant color that was in the annotation. Clusters did not appear to have very distinct classifications, and many of the groupings appeared identical.

When k = 2, the clusters created are closely related. Above displays subsets of the 2 clusters.

K = 6:9

Increasing the amount of clusters did not drastically change the types of groupings we got, but also did not appear to create distinct grouping types. Some groups contained more artists than others, but all groups still contained artist notations. Increasing the cluster size appeared to intermix the more distinct grouping types, instead of finding a new shape classification.



K = 8 clusters with an increased subset showing 49 samples for each cluster.

K = 10

When we attempted to create clusters with anything over k = 10, we encountered a memory error: "Insufficient Java memory to create the output.". For this reason, we decided to not move beyond testing k = 10, and because k = 9 was not improving our cluster groupings.

V. Discussion and Future Work

Interpretation of Clusters:

The artist cluster we originally set out to discover was very elusive. Creating larger amounts of clusters appeared to help narrow down possible artist clusters, but there was still significant overlap in the types of annotations each grouping had. The more accurate groupings kmeans created were groupings that were predominantly one color. Other groupings we saw were clusters where we believe the user followed the shape hierarchy, but also classified a larger amount of the shape as pink, green, and/or red.

Final Clusters:

Within the folder "our_clusters" you can see examples of clusters subsets that kmeans created. The images are labeled k#-cluster#.png with the k# showing what k equals, and the cluster# showing which cluster is being displayed.

Potential Issues and Future Work:

We considered one of the reasons our clusters were lacking a distinct artist grouping was because our features were not sufficient for finding artists in the dataset. In future work, we would like to explore more features we can generate and see if they would be better at grouping artists together. One way to do this might be to incorporate the golden annotations that were included in the data. We did not use the golden annotations in this project because the golden annotations are only for a subset of the shapes, and we didn't want to generalize the meaning of these very specific annotations across all the shapes. Another thing we would like to explore in future work is how our clusters change based on the distance metric we use for K Means Clustering. For this project we only used 'sqeuclidean'.