CSCI E-181 Spring 2014 Practical 1

David Wihl
davidwihl@gmail.com

February 12, 2014

Warm-Up

As a warmup, I synthesized five clusters of data. I then used a K-Means implementation in Octave I had written for a previous course.[1] While this implementation was sufficient for the prior course's provided dataset, when I tested it the synthesized data set, K=5 and random initial centroids, one of the centroids would frequently not converge on any points.

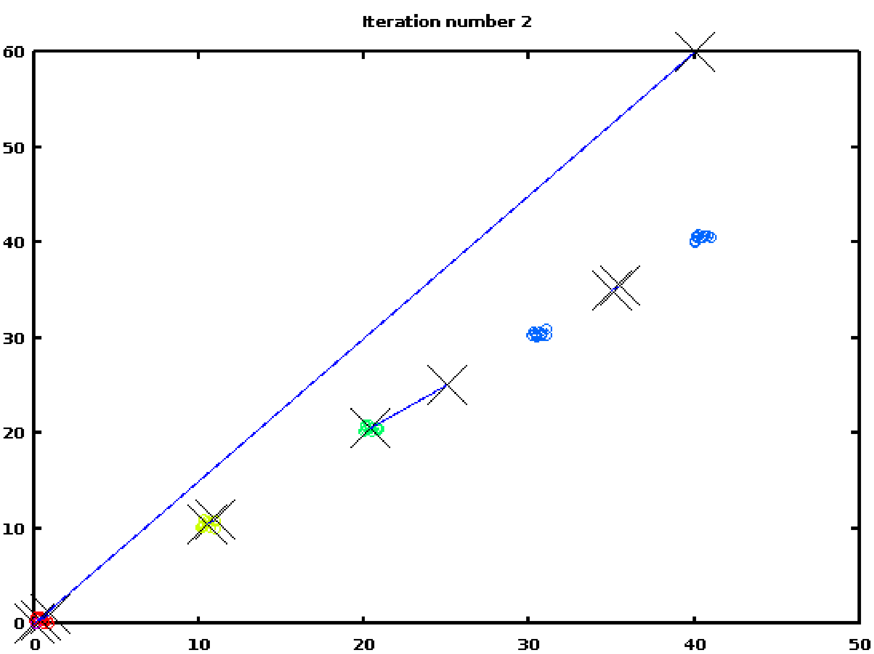

Figure 1: Random Initial Centroids After 1 Iteration

[1]Machine Learning, Coursera, Prof. Andrew Ng, Completed Jan 2014, `https://class.coursera.org/ml-004`

I subsequently modified the code to use K-Medoids, choosing one of the sample data points at random as an initial centroid. This worked much better.

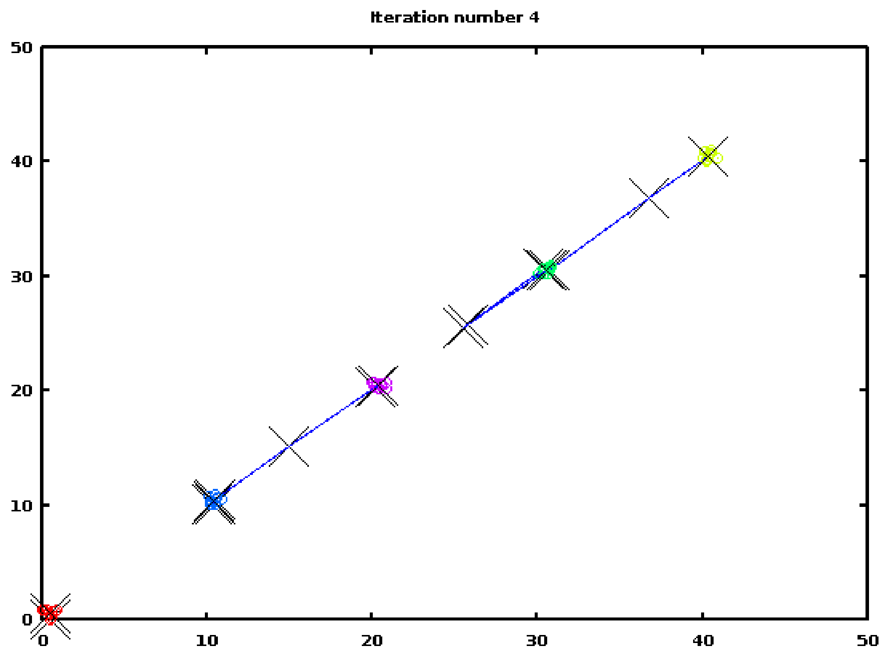

Figure 2: K-Medoids Converge After 4 Iterations

CIFAR-10 Image Data

I then attempted using K-Medoids with the CIFAR-10 Image Data, using the Matlab version of the data with Octave. The training data consists of a 10000x3072 matrix of UInt8. Each row is a 32x32x3 (total 3072 columns) color image, consisting of 1024 red, 1024 green and 1024 blue elements. There are 10 classes in the set ("airplane", "automobile", etc.), so setting K=10 was a rational first step.

Percentage Distribution of K values after normalization and 10 iterations

06 05 04 26 14 13 05 04 03 15

TODO: fill this in

Recommender System

For the main part of the exercise, I investigated a series of increasing complex algorithms.

Pearson Distance

The first was using Pearson distance from *Programming Collective Intelligence*.[2]

$$r = \frac{\sum_{i=1}^{n} x_i y_i - \frac{\sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n}}{\sqrt{\sum_{i=1}^{n} x_i^2 - \frac{(\sum_{i=1}^{n} x_i)^2}{n}} \sqrt{\sum_{i=1}^{n} y_i^2 - \frac{(\sum_{i=1}^{n} y_i)^2}{n}}}$$

Figure 3: Pearson Correlation Coefficient Approximation

Unfortunately Pearson distance

[2]Programming Collective Intelligence by Toby Segaran. © 2007 Toby Segaran, 978-0-596-52932-1.