Keaton Spiller

CS 445

Winter 2022

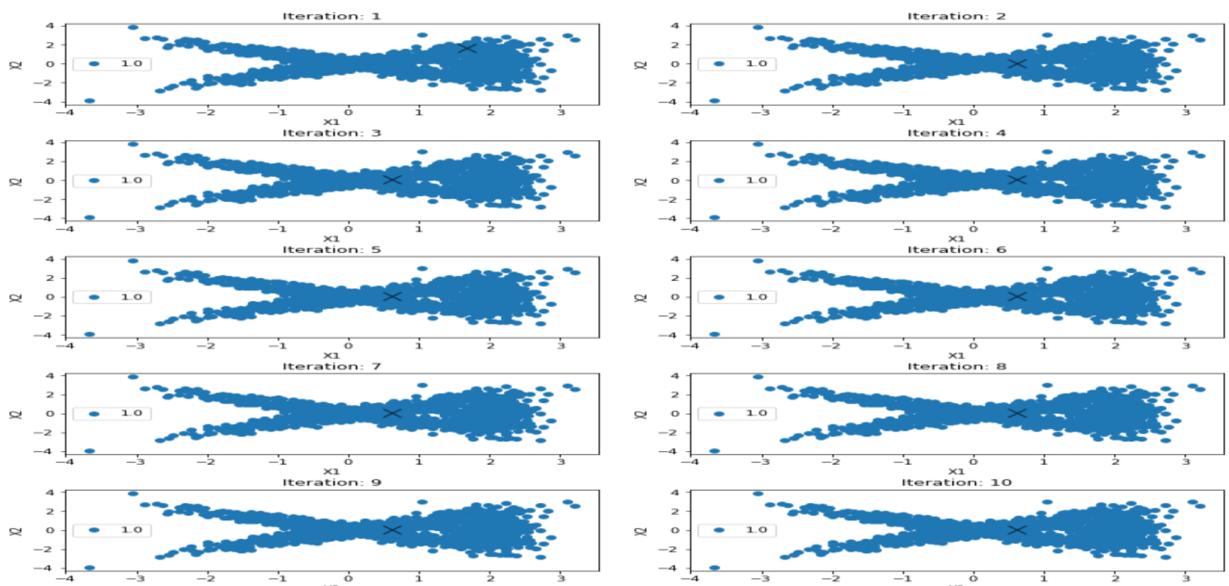
Assignment 3

Incorporating the algorithm on hard K-means was significantly easier than fuzzy c means, and without the use of sci-kit tools was a significant challenge. For the first and second half I needed to figure how to split up the centroid operations between the x1 and the x2 coordinates in order to L2 normalize the data without losing it's point wise multiplication. For the fuzzy c_means I needed to preserve the vector on the top of the c_k, and avoid storing the cluster values in index's . The fuzzy c_means was able to cluster and find patterns beyond what the hard K means clustering could, leading to some really interesting mixtures of clusters, beyond what I could piece together by hand.

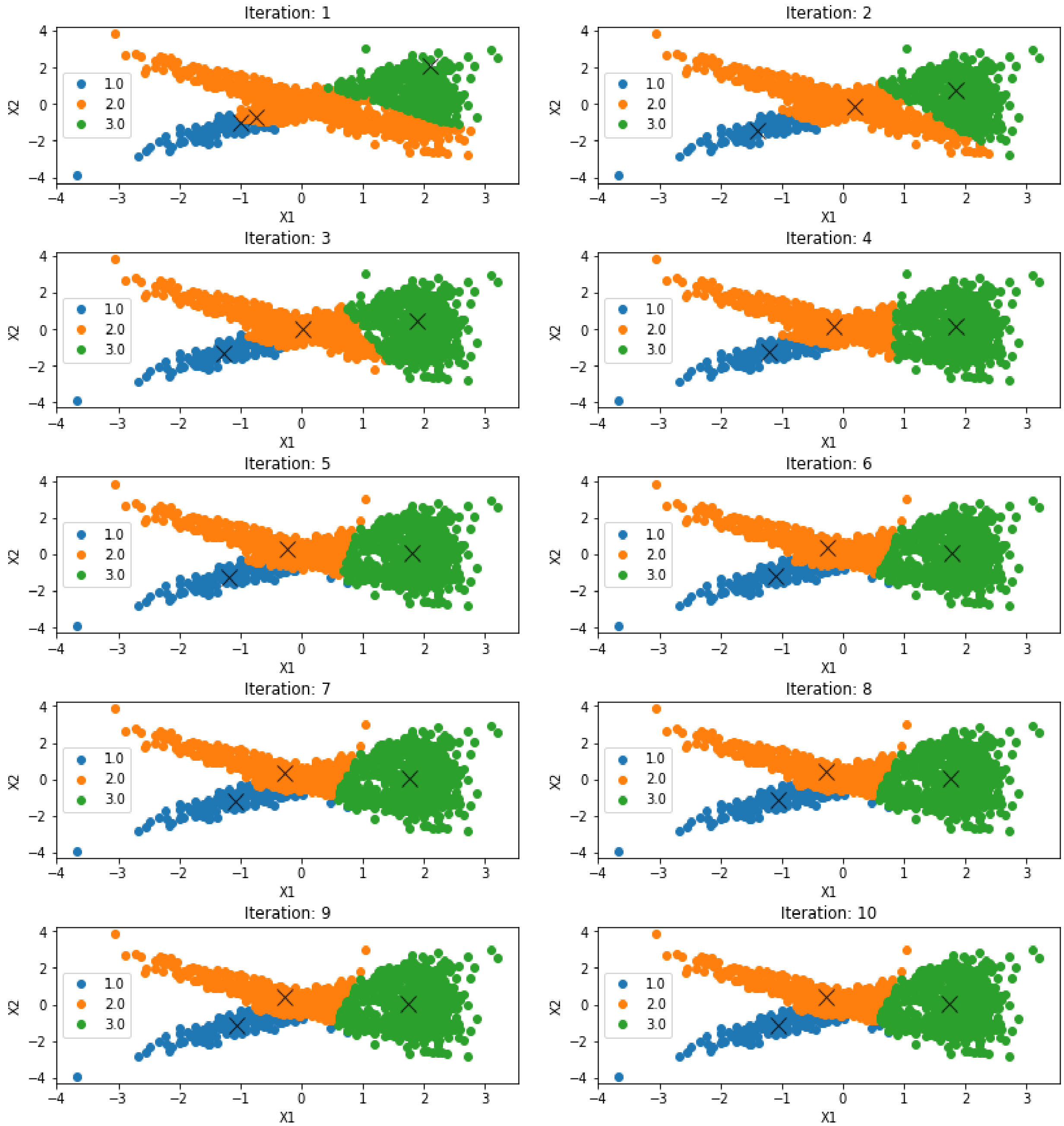
(Cluster_x1_k = m_step_topx1[:, k, iteration] / bottom )

**Assignment #1: K-Means**

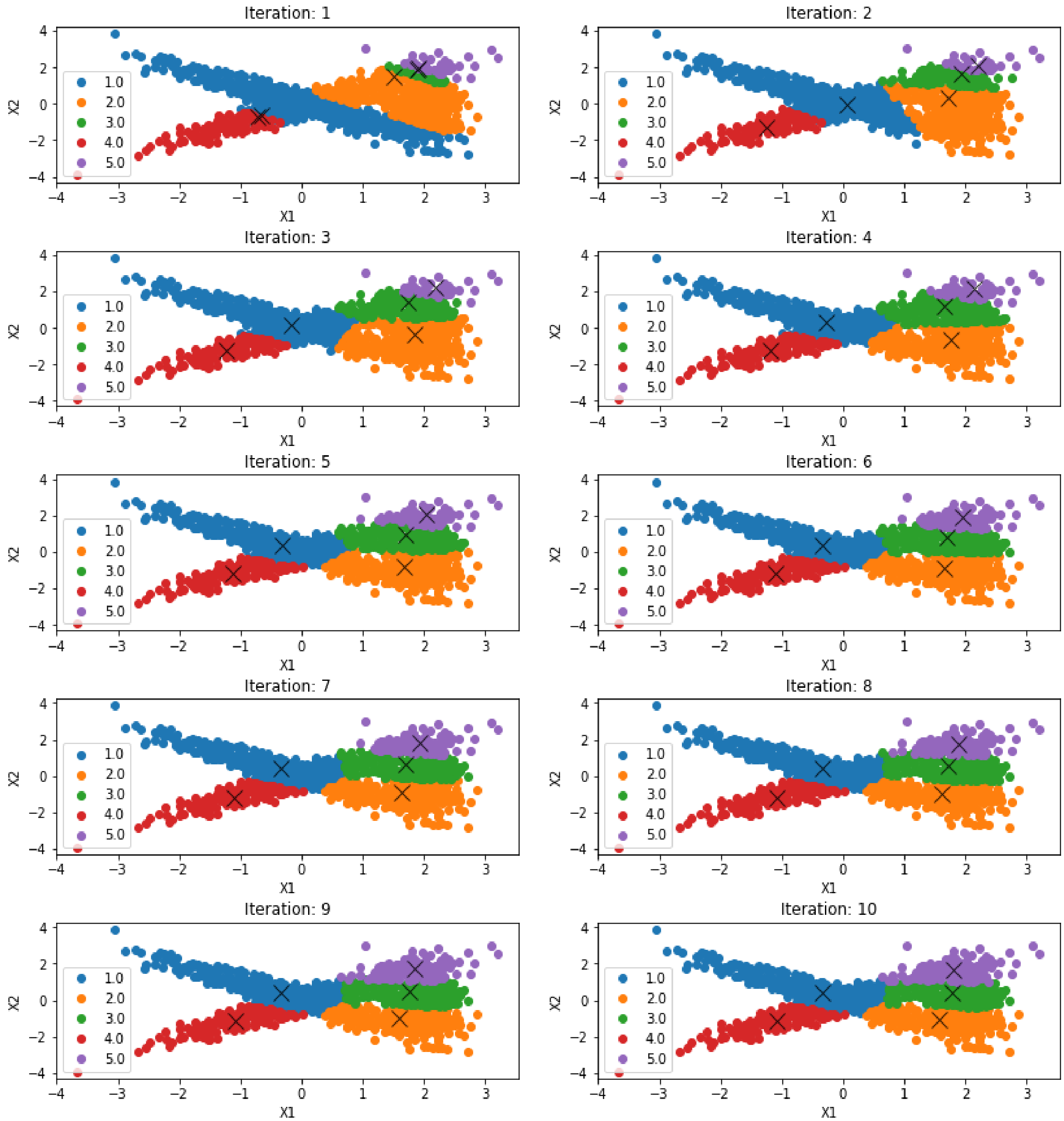
**For 10 iterations r=10**




K = 1, best iteration = 2, SSE = 2186.86898371



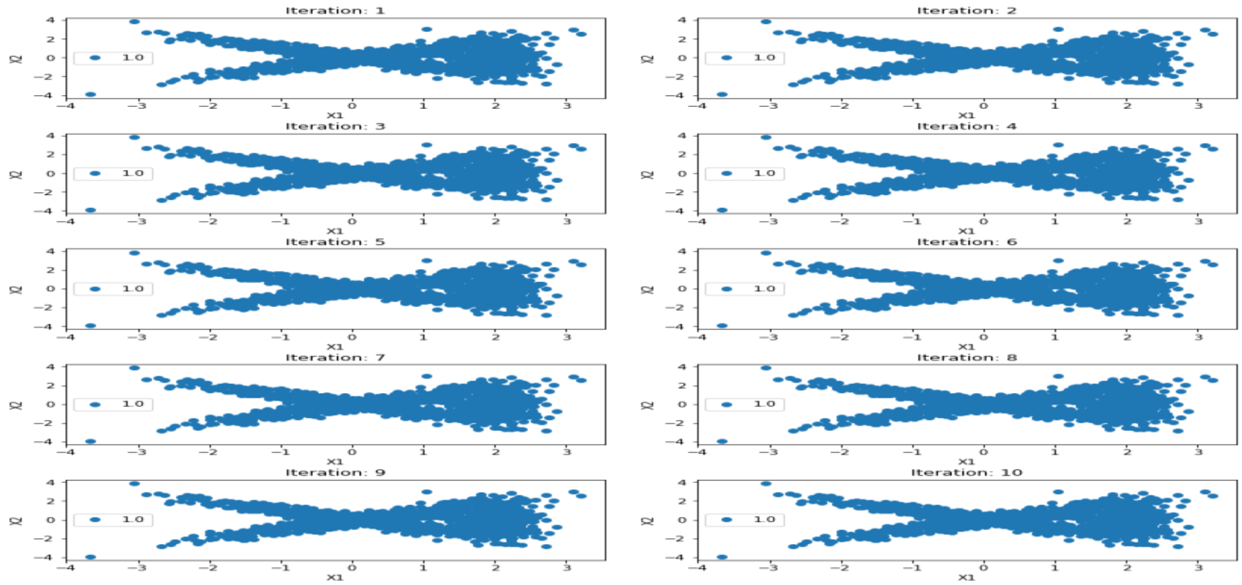

K = 3, Best Iteration = 10, SSE = 8182.1250289



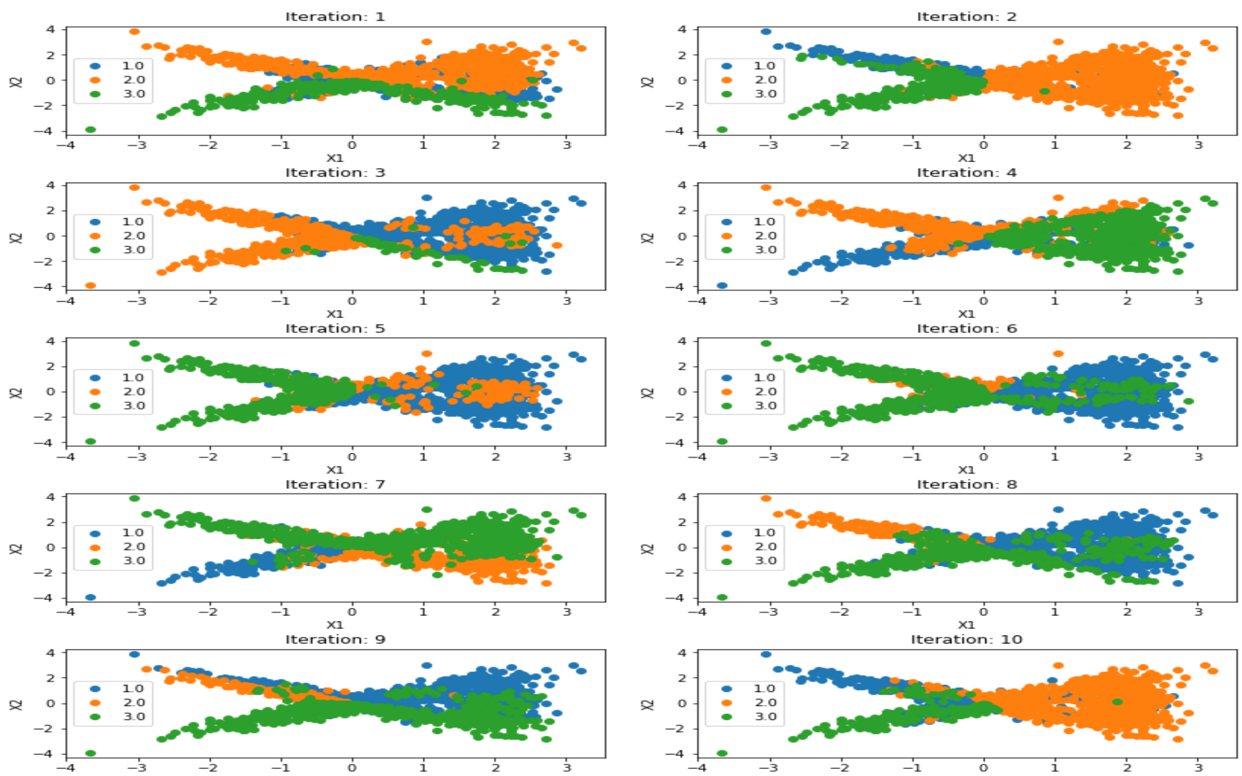

K = 5, Best Iteration : 10, SSE = 15318.21418594

It's pretty amazing that within 3 iterations the k-means algorithm is able to come up with distinct separations between the clusters

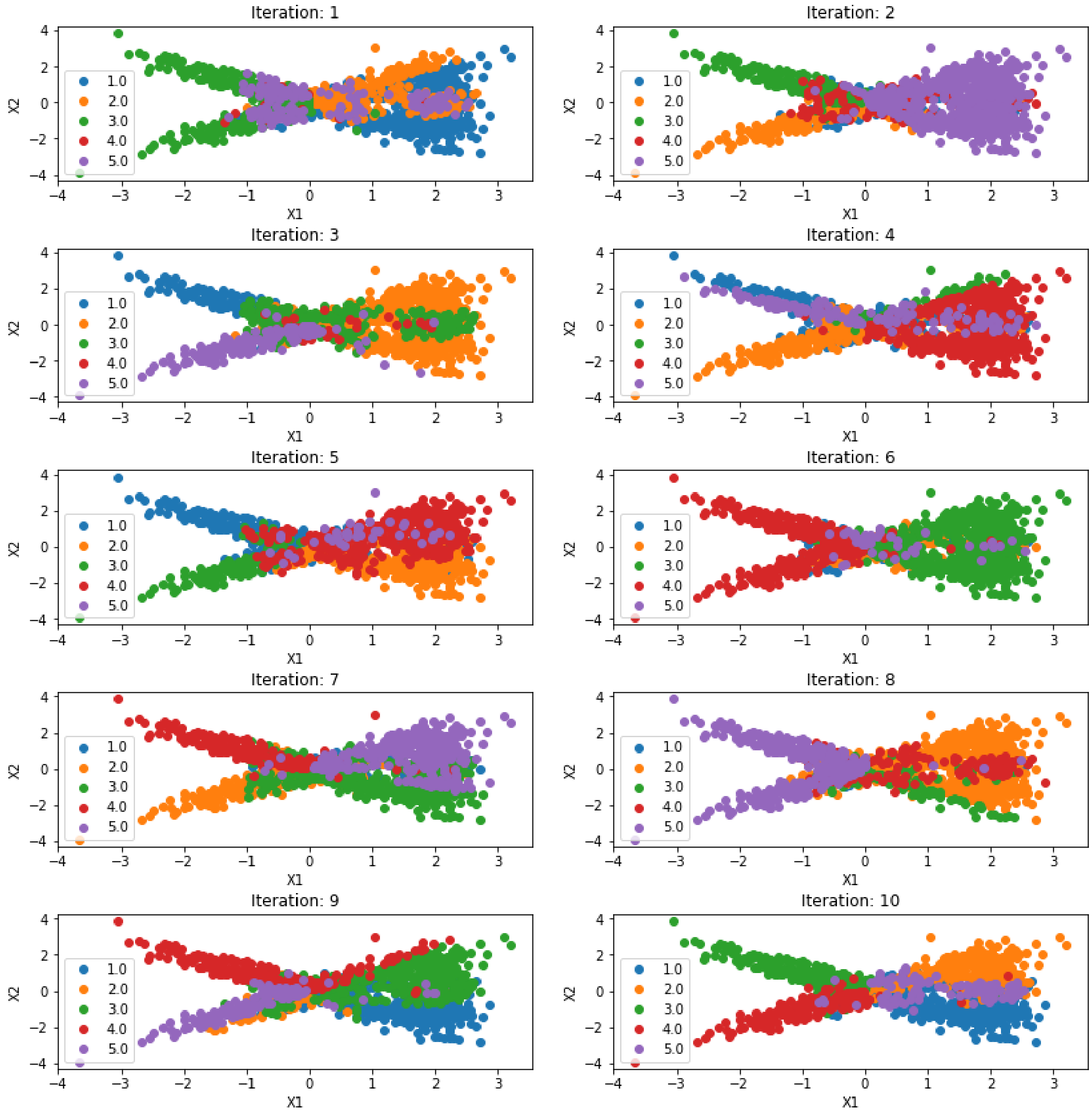
## Assignment #2: Fuzzy C-Means




For 10 iterations r=10, m=2, C = 1 , best iteration=1, SSE = 49.63987632




For 10 iterations r=10, m=2, C = 3, best iteration 4, SSE = 177.12669922




For 10 iterations r=10, m=2, C = 5, best iteration 4, SSE = 321.49255501

Visually iteration 10 looks the best, but because of how mixed the cluster's are this feels harder to interpret than hard K-Means.