CS 383 - Machine Learning

Assignment 4 - Clustering

Introduction

In this assignment you'll work on clustering.

You may not use any functions from machine learning library in your code, however you may use statistical functions. For example, if available you **MAY NOT** use functions like

- k-mean

Unless explicitly told to do so. But you **MAY** use basic statistical functions like:

- std
- mean
- cov
- eig
- svd

Grading

Part 1 (Theory)	30pts
Part 2 (Clustering)	70pts
Part 3 (Extra Credit)	20pts
TOTAL	100pts

Table 1: Grading Rubric

DataSets

Pima Indians Diabetes Data Set In this dataset of 768 instances of testing Pima Indians for diabetes each row has the following information (1 class label, 8 features).

0. Class Label (-1=negative,+1=positive)
1. Number of times pregnant
2. Plasma glucose concentration
3. Diastolic blood pressure (mm Hg)
4. Triceps skin fold thinkness (mm)
5. Insulin (mu U/ml)
6. Body mass index (kg/m^2)
7. Diabetes pedigree function
8. Age (yrs)

Data obtained from: https://archive.ics.uci.edu/ml/support/diabetes

1 Theory

1. (3pts each = 12pts) Given two clusters:

$$C_1 = \{(1,2),(0,-1)\}, C_2 = \{(0,0),(1,1)\}$$

what is:

   (a) The weighted average intra-cluster distance if you are using euclidian distance?

   (b) The single link similarity between the clusters if we're using cosine similarity as our similarity function?

   (c) The complete link similarity between the clusters if we're using cosine similarity as our similarity function?

   (d) The average link similarity between the clusters if we're using cosine similarity as our similarity function?

2. (10pts) Given an average intracluster distance for clustering level j, W_j, what is the **fourth** derivative at j, namely W_j''''?.

3. (8pts) Given the output of your clustering algorithm as $C_1 = \{1,2,3,4\}, C_2 = \{5,6,7,8\}$, and a hand labeled clustering of $C_1 = \{3,4\}, C_2 = \{1,2,5,6,7,8\}$, what is the weighed average purity of the clusters created by the clustering algorithm?

2 Clustering

Let's implement our own version of k-means to cluster data!

Write a script that:

1. Reads in the data. For demonstrative purposes, you will be using the dataset mentioned in Datasets section (i.e. the diabetes dataset). However, given another dataset's observable data matrix, X, your code should still work.
2. Separates the class label from the observable data to form matrices Y and X, respectively.
3. Standardizes the features.
4. Passes the observable data X and the class labels Y to a (user-created) function called *myK-Means* that takes three parameters:
   (a) The data to cluster, X
   (b) The target clustering, Y (we'll use this for measuring purity as wel go)
   (c) The number of clusters, k

The myKMeans function :
Your *myKMeans* function should run k-means on the supplied data and value of k. Since we'll visualize the clustering process, a few caveats:

- Although theoretically your code should work for any value of $1 <= k <= N$, we'll constrain k to have a maximum value of 7, so that you will only need to define up to 7 colors.
- Although your code should be written in such a way to be able to cluster in any dimensional space, for visualization purposes, you may assume dimesonality $D > 1$, and if your data's dimenionality is greater than 3, first reduce its dimensionality to 3 using PCA. You may use the code you developed in HW3 to do this, or a PCA function from your linear algebra library.

In order to visualize the cluster process, you have to generate plots of initial and terminal clustering. For these plots you should:

1. Display the current reference vectors as solid circles and the observations as x's.
2. The reference vectors and associated observations should have the same colors. For instance, if you use blue to represent cluster 1, then its reference vector should be a solid blue circle, and all observations associated with it should be blue x's.
3. At the top of your graph you should state the iteration number as well as the current weighed average purity of the clusters, using the vector Y as the hand-labeled cluster assignments.

Figures 1-2 show the an example of initial and terminal clusterings when $k = 2$.

Implementation Details

1. For reproducability and being able to compare results to others, it is recommended that you seed your random number generator before using it. I typically seed mine with zero.

2. You can decide how you want to initialize your reference vectors. Refer to the slides for ideas.

3. Use the L2 distance (Euclidean) to measure the distance between observations and reference vectors.

4. Terminate the training process when the sum of magnitude of change of the cluster centers (from the previous iteration to the current one) is less than $\epsilon = 2^{-23}$. That is, when $\sum_{i=1}^{k} d(a_i(t-1), a_i(t)) < \epsilon$ where k is the number of clusters, $a_i(t)$ is the reference vector for cluster i at time t and $d(x, y)$ is the L1 distance (Manhattan) between vectors x and y (as defined in the *Similarity and Distance Functions* link on BBlearn).

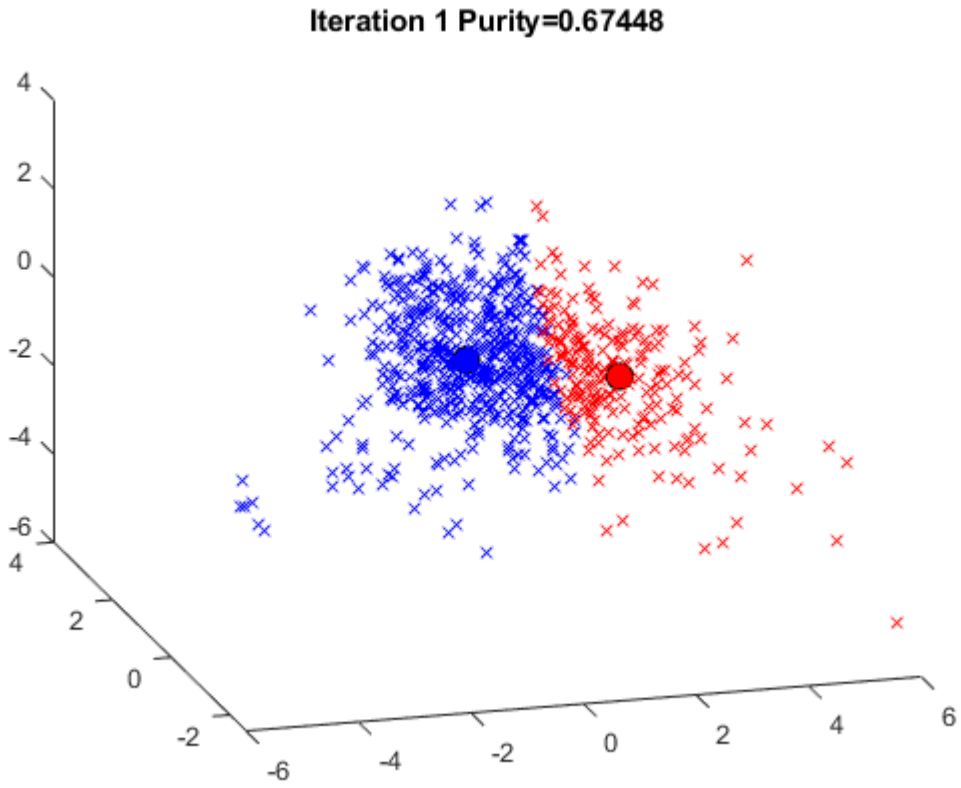

Figure 1: Initial Clustering

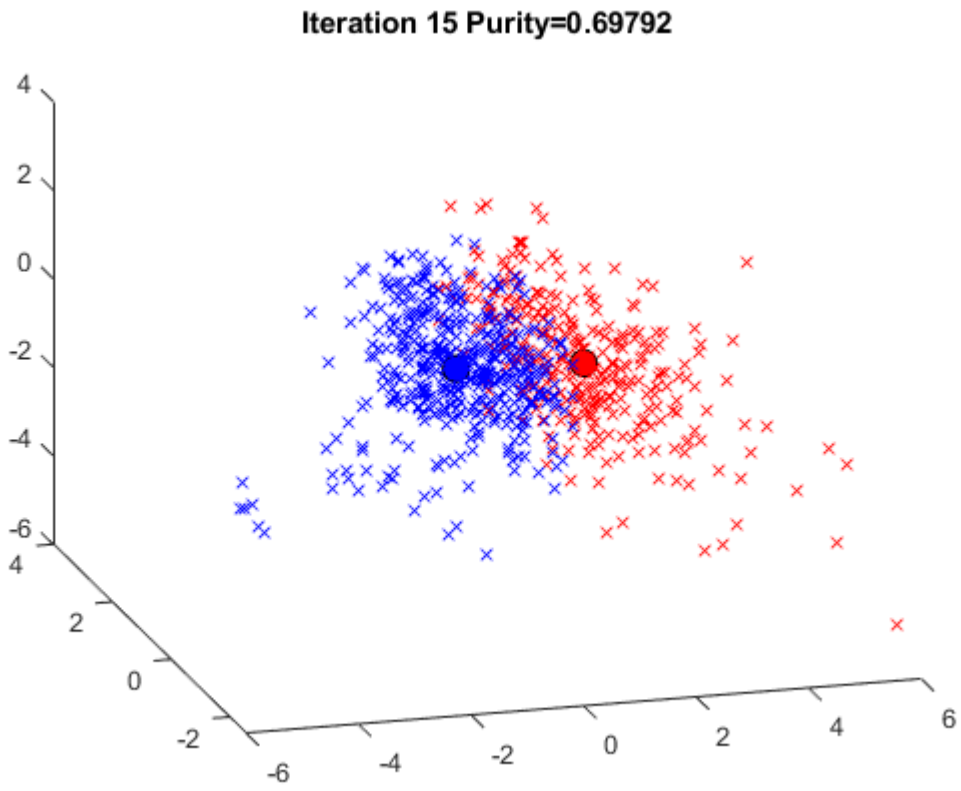

Figure 2: Final Clustering

3 Extra credit

In order to better visualize the cluster process, for extra credit, you'll generate a video. Choose a frame rate that is natural for observing the changes in the association. Each frame of the video will be a plot showing the current clustering. If X has only 2 features, this will be a 2D plot. If it has 3 features (either natively, or after PCA reduction), then this will be a 3D plot. If possible (that is, if the graphics library you are using allows it), try to have any 3D plots rotated so that it is easier to see what's going on.

Python Note: You may want to consider using opencv's Python module for generating your video. To install this on your machine (or locally on tux) you can run the command
pip3 install –user opencv-python.

Submission

For your submission, upload to Blackboard a single zip file containing:

1. A LaTeX typeset PDF or Jupyter Notebook PDF containing:
   (a) Part 1: Your answers to the theory questions.
   (b) Part 2: The visualization of k-means cluster centers, plots of initial and terminal clustering.
   (c) part 3 (extra credit): Sample videos of at least three different runs with different values of k. The video filenames should be in the format

   $$K\_[k].[ext]$$

   where:
      (a) [k] is the value of k passed to your function.
      (d) [ext] is the file extension.
   (d) Source Code - python notebook