EE141L: Lab 6
Distance based clustering and classification

TBD

1 Clustering and classification

1.1 Clustering

In machine learning applications the goal is to use data to learn how to automate certain tasks, e.g., image classification, identity verification, disease risk estimation, stock market prediction, etc. A key tool for working with data is *Clustering* also known as *unsupervised learning* or *community detection*. A *cluster* is defined as a group of similar objects, using an a similarity metric appropriate for the application at hand. Clustering then refers to algorithms and techniques to group (or cluster) objects according to their similarity. Examples include

1. *Community detection in social networks.* Users form clusters based on political view, socio-economic background, location, age, etc.

2. *Computer vision.* Images can be grouped according to similarity, e.g., to search images representing specific objects on the Internet. Video analysis can be used for pedestrian/car detection in self driving cars.

3. Genetics[1]: grouping genes according to common characteristics.

4. Marketing: group consumers according to behavior and spending habits. Identify new trends by detecting emerging new communities.

Clustering presents several major challenges

- We may not know *a priori* how many clusters there will be.

- Similarity may be subjective or it may be difficult to find a reliable metric to quantify similarity.

- Clusters may evolve over time.

[1]Novembre, John, et al. "Genes mirror geography within Europe." Nature 56.7218 (2008): 98-101. APA

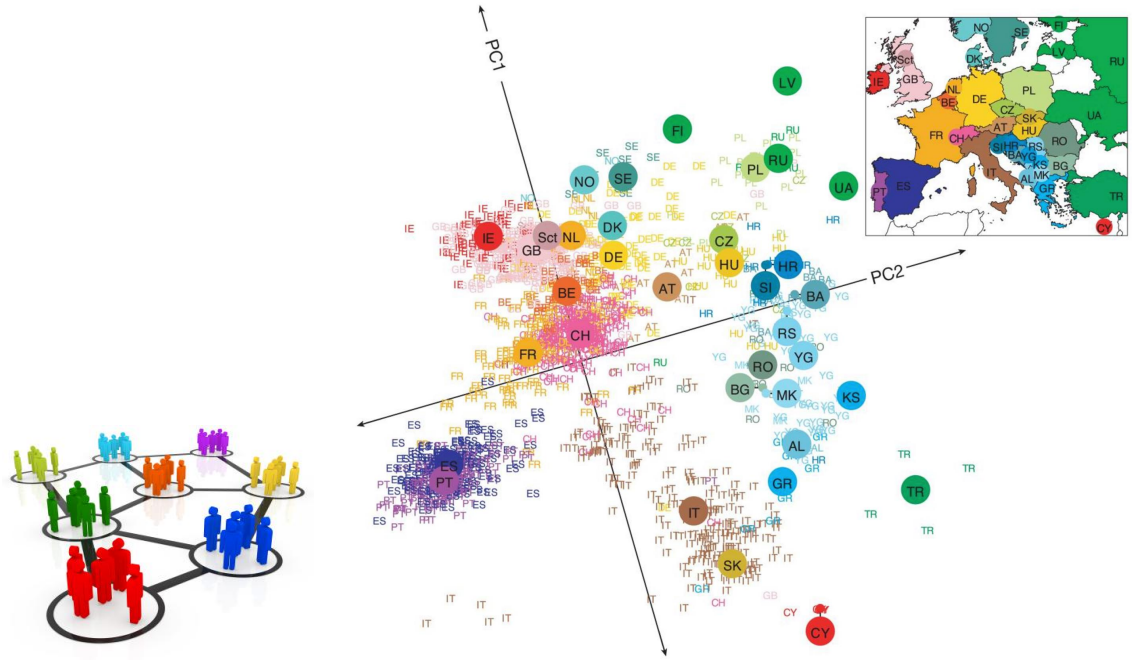

Figure 1: (Left)Communities in a social network. `https://www.ignitesocialmedia.com/social-networks/`. (Right) Genes mirror geography within europe

1.2 Classification

Clustering can also be interpreted as a method to understand data. Once the clusters are known, they can be used to process new data that comes in. For example:

- When a new picture is uploaded to Facebook face detection algorithms are used to make it easier for users to tag their friends in those photos. To achieve this it was necessary to develop methods that would analyze an image and can classify certain sets of pixels as corresponding to face or other object. Then a second classifier, suggests friend's names.
- When a new user joins Netflix, some data is collected about a few shows/movies the user has liked/watched, so that the system can start making suggestions. These suggestions are based on finding clusters of other users with similar preferences to those of the new user.

2 In this Lab

2.1 Motivation

The goal of this lab is to show how the *inner product*, *norm*, *distance* we have discussed in class, which are quite intuitive when applied to 3D space, can be applied to much higher dimensional spaces and still provide some very useful notion of similarity. In particular we focus on images, which are sets of 28×28 pixels and thus can be viewed as vectors in the

space $M(28, 28)$ of dimension 784. Clearly, it is no easy thing to think geometrically in such a space, but we will see how distances between objects (individual images) can be used to cluster images of this size, capturing their similarity. We will use the same metrics defined in class, that is, the norm of a vector in $\mathbf{x} \in \mathbb{R}^n$ is $|\mathbf{x}|$ defined as

$$|\mathbf{x}|^2 = x_1^2 + x_2^2 + \ldots + x_{n-1}^2 + x_n^2$$

and the distance between two vectors in the space is the norm of the difference vector, so that

$$d(\mathbf{x}, \mathbf{c}) = \sqrt{(x_1 - c_1)^2 + (x_2 - c_2)^2 + \ldots + (x_{n-1} - c_{n-1})^2 + (x_n - c_n)^2}.$$

3 Dataset

For this lab, we will be working with the MNIST dataset, which is a database of handwritten digits. This database is one of the most common datasets used for image classification. The digits have been size-normalized and centered in a fixed-size image (28×28 pixels).The MNIST dataset is useful for people who want to try learning techniques and pattern recognition methods on real-world data while spending minimal efforts on formatting and preprocessing.

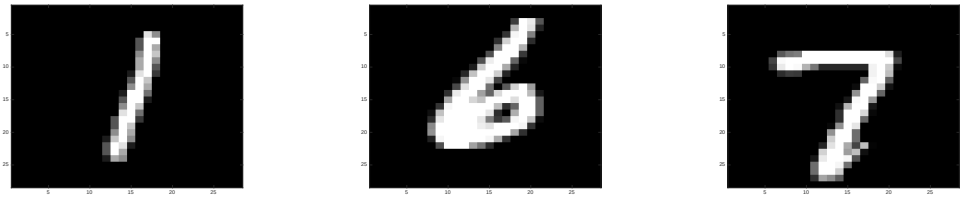

Figure 2: Three images in MNIST dataset.

3.1 Problems

You are given a set of images, in the first part you will cluster them into 3 groups. In the second part you will use these clusters to classify new images. You will use for clustering an algorithm called k-means. Let $\{\mathbf{x}_i\}_{i=1}^N$ be a set of vectors in $\mathbb{R}^n$ (here we are working in $\mathbb{R}^{784}$!!). We want to find k clusters, which is equivalent to finding k vectors $\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_k$ that represent well each cluster. The k-means algorithm works as follows (see pseudocode, and animation `http://shabal.in/visuals/kmeans/1.html`)

Problem 1. Start matlab run_kmeans.m script and load the .mat file mnist_train.mat. Each column of the variable data corresponds to a vectorized 28×28 image. There are three

Algorithm 1 k-means clustering

Require: initial centroids $\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_k$.

while not converged **do**
  for each $\mathbf{x}_i$ **do**
    compute distances $d(\mathbf{x}_i, \mathbf{c}_1), d(\mathbf{x}_i, \mathbf{c}_2), \cdots, d(\mathbf{x}_i, \mathbf{c}_k)$
    Compute closest cluster to point $\mathbf{x}_i$ $\min_j d(\mathbf{x}_i, \mathbf{c}_j)$
    Assign point $\mathbf{x}_i$ to its closest cluster
  end for
  Now that all points have been assigned a cluster, update $\mathbf{c}_j$ as average of all points in cluster j.
end while

different types of images, use the functions imshow and reshape and report one instance of the three types. For example you can display the first image using imshow(reshape(data(:,1),28,28))

Problem 2. Run k-means as follows [idx, C] = kmeans(data,3,'replicates',5);. This code will run k-means five times, and pick the best result. The vector idx will have values in $\{1, 2, 3\}$, so if idx(i)==j, that means the data point data(i,:) is assigned to cluster j.

1. Report the $\mathbf{c}_j$ as images.

2. When you load the .mat file, you also have access to the true labels (in the variable labels), in that variable there are also 3 different values, one corresponding to each cluster. You will have to manually identify, which variable value in $\{1, 2, 3\}$ corresponds to each cluster in the variable labels. Then compute error

$$\text{error} = \frac{\#\text{points in correct cluster}}{\#\text{points}}. \tag{1}$$

3. Give 3 examples, one for each class, where the assigned cluster is wrong. Report image, image index (which row of data), and assigned cluster.

Problem 3. Now you are given new data in the mat file mnist_test.mat. You will implement a classifier using the clustering from the previous problem. Each row in the matrix data_test is a point to be classified.

1. Let's call the i-th point by $\mathbf{y}_i$, then implement the following

Algorithm 2 classification

Require: Class centroids $\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_k$, new data $\{\mathbf{y}_i\}_{i=1}^M$.

for each $\mathbf{y}_i$ **do**
  compute distances $d(\mathbf{y}_i, \mathbf{c}_1), d(\mathbf{y}_i, \mathbf{c}_2), \cdots, d(\mathbf{y}_i, \mathbf{c}_k)$
  Compute closest cluster to point $\mathbf{y}_i$ $\min_j d(\mathbf{y}_i, \mathbf{c}_j)$
  Assign point $\mathbf{y}_i$ to its closest cluster
end for

To compute the distance between vectors **a** and **b** you can use the matlab function norm(a-b). As a suggestion, in Step 2, you can form the vector $[d(\mathbf{y}_i, \mathbf{c}_1), d(\mathbf{y}_i, \mathbf{c}_2), d(\mathbf{y}_i, \mathbf{c}_3)]$. Then in Step 3, you can use the Matlab function min.

2. Report error

3. Give 3 examples of wrongly classified images, one for each class.

Problem 4. – Discussion

1. What is the minimum number of clusters one should choose?

2. How would you use distance to determine how similar digits are to each other?

3. How would you use the result to determine which of the digits have more variation in terms of how they are written?