CISC 271 Class 19

Unsupervised Learning – K-Means Clustering

Text Correspondence: B &V 4.3–4.5

Main Concepts:

- *Unsupervised classification*
- *Cluster: a set with a centroid*
- *K-means: finding* k *clusters of data vectors*

Sample Problem, Machine Inference: How can we find and test automatic clustering of data?

19.1 Supervised and Unsupervised Classification

Classification is a partition of a set of data into distinct subsets. Classification problems can be roughly divided into *supervised* classification, in which some of the data have an attribute or label that distinguishes them, and *unsupervised* classification. In this class we will briefly explore unsupervised classification, which is usually called *clustering*.

19.2 Unsupervised Classification: Data Clustering

One kind of unsupervised classification is cluster analysis, or clustering. Each subset of data is referred to as a *cluster*. The data in a cluster are a "natural group", which is often defined by some measure of the data. Depending on the domain-specific definition of a cluster, two cluster may be distinct – having no data in common – or they may overlap – the same data may be present in more than one cluster.

For us, clustering will be the partitioning of data into two or more distinct subsets. The data will be vectors that all have the same number of entries, so these vectors "live" in a vector space. The data we will use as example will sometimes have a non-numerical attribute that can be used to assess the success of a clustering algorithm.

Two frequently encountered definitions are: a cluster is a set of data; and, a cluster is a representative data member that has neighbors. Our data are vectors $\vec{x}_j$, and the neighbor relationship is often defined by using a vector norm $\|\cdot\|$. We can formalize these two definitions with these conventions.

Definition: Cluster as a set

For any non-empty finite data set $X \in \mathbb{R}^m$ that has m members $\vec{x}_j$, a cluster S_i is a set with $m_i > 0$ members that is part of a partition of X so that

$$S_i \subset X \tag{19.1}$$

Definition: Cluster as a centroid

For any non-empty finite data set $X \in \mathbb{R}^m$ that has m members $\vec{x}_j$, a cluster S_i is the set with a centroid $\vec{g}_i \in X$ such that, for any distinct centroid $\vec{g}_k \in X : \vec{g}_k \neq \vec{g}_i$,

$$S_i = \{\vec{u} \in X : \|\vec{u} - \vec{g}_i\| < \|\vec{u} - \vec{g}_k\|\} \tag{19.2}$$

Observations: Definition 19.1 has a centroid that can be computed from its members. Definition 19.2 is a computation of a cluster set, with the caution that a "tie-breaking" rule may be required to produce non-trivial clusters that have more than one member.

Clustering is one form of unsupervised learning, in which a computer program does not use additional information to perform the learning task. These tasks are generally hard, so we will use clustering as a way to understand both an important data set and some of the mathematical concepts that are fundamental in current machine learning.

19.3 Clustering – Iris Data

We will begin by considering one of the earliest data sets that is still in use. It was described by Sir Ronald Fisher, who is a founder of both statistics and genetics. His data [5], gathered from the Gaspé Peninsula in Quebec, were measurements of 150 Iris flowers. We will use two of the measurements, which are the length and width of the sepals or flower coverings, so we have 150 vectors of size 2. The additional attribute is the species; we will refer to these as B type (Beachhead Iris, *I. setosa*) and P type (Purple Iris, either *I. virginica* or *I. versicolor*).

Our first action is to graphically display the vectors without the species attribute. Figure 19.1 shows the 150 data vectors as distinct asterisk symbols.

Visually, the data seem to form two clusters, with the lower-left being the B type and the upper-right being the P type. This data set is useful in learning about types of clustering algorithms, types of supervised classification algorithms, and many other aspects of machine learning.

For the Iris data set, we are interested in finding the two clusters that correspond to the B type of plant and the P type of plant. We thus want to find $\vec{g}_1$ and $\vec{g}_2$ that distinguish set S_1 from set S_2.

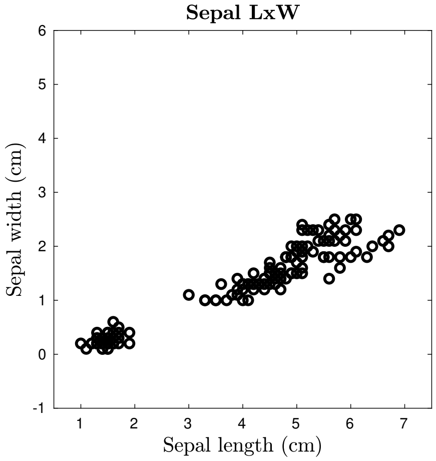

Figure 19.1: Fisher's Iris data set. The sepal length (in centimeters) is the horizontal axis and the sepal width (in centimeters) is the vertical axis, for the 150 samples in the data set.

19.4 K-Means Clustering

In machine learning, the clustering problem is often phrased in terms of finding k means, or centroids, that partition the data into k sets [8]. This is called the *k-means* clustering problem, with a long history dating to at least 1957. There is a well known algorithm, originating in Bell Labs, that was developed by Stuart Lloyd [7] to heuristically approximate a solution to finding the means $\vec{g}_i$ of Definition 19.2. MacQueen's algorithm is so widely used that it is usually referred to as *the* k-means algorithm. It is relatively simple to implement, with certain subtleties regarding how to initialize the algorithm, but we will not delve further into the implementation details.

19.4.1 A K-Means Algorithm

The idea that underies the basic k-means algorithm is quite simple: we alternate between Definition 19.1 of a cluster and Definition 19.2 of a cluster. There are two ways to write a basic k-means algorithm, which depend on how we intialize the clusters.

The first method, which is seldom implemented, begins with a partitioning of the data that satisfies Definition 19.1 and iteratively updates the clustering.

In pseudocode, we can write this algorithm as

```
Randomly initialize $k$ partitions of $X$ as $S_i$
while ~converged
    for each index $i$:
        compute $\vec{g}_i$ of data $\vec{x}_j$ that have index $i$
    for each $\vec{x}_j \in X$:
        assign index $i$ of nearest $\vec{g}_i$
```

The second method, which is commonly implemented, begins with a selection of centroids of the data that satisfies Definition 19.2 and iteratively updates the clustering. In pseudocode, we can write this algorithm as

```
Randomly initialize $k$ centroids $\vec{g}_i \in X$
while ~converged
    for each $\vec{x}_j \in X$:
        assign index $i$ of nearest $\vec{g}_i$
    for each index $i$:
        compute $\vec{g}_i$ of data $\vec{x}_j$ that have index $i$
```

We can see that the iterations of these two methods are essentially the same. The difference is how the iterations are initialized.

Convergence can be measured in many ways. One measure is that the centroids are insignificantly changed by an iteration. Another measure is that the indexes are unchanged by an iteration. The first measure requires a threshold for significance and the second method requires comparisons for each data vector $\vec{x}_j \in X$. Additionally, a limit may be imposed on the number of iterations that are computed.

19.4.2 Example: Fisher's Iris Data

The k-means algorithm is currently available in the general Matlab distribution as the function `kmeans`. To perform k-means clustering on the Iris data in Matlab, we can:

- Load the data from the Matlab repository
- Extract the relevant portions as a 150×2 matrix
- Perform k-means clustering with $k = 2$

Part of the instructor's Matlab code for this is in the extra notes for this class. The code finds a 2×2 matrix, with the first column vector being the centroid of the type-B Iris and the second column vector being the centroid of the type-P Iris.

These are displayed graphically in Figure 19.2, with type B shown as small circles and type P shown as small crosses. The centroid of each cluster in Figure 19.2 is shown as a large "X".

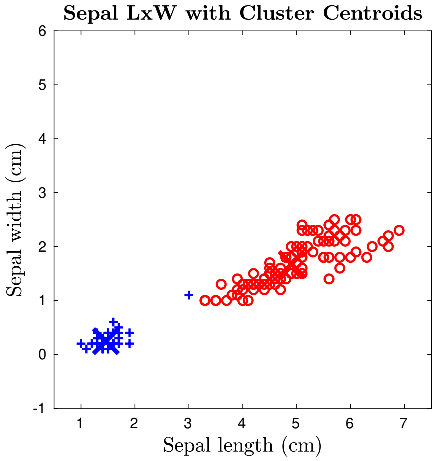

Figure 19.2: Fisher's Iris data set, clustered using the k-means function in Matlab. The first cluster is shown as circles and the second cluster shown as crosses; each centroid is shown as a large "X".

Visually, the centroids from the k-means algorithm seem to accord well with the clusters that a human might compose. How well does the algorithm perform on this data set?

Recall the distinction between training and testing that we first drew when we studied the problem of finding linear patterns in sparse data. Here, the process of training is the computation of the two means $\vec{g}_1$ and $\vec{g}_2$. The process of testing is the determination of the performance of the k-means algorithm on a particular dataset.

In general, this testing process is a difficult problem in machine learning. For our specific Iris data set, we have a valuable data attribute: each data vector was labeled, by a human expert, as a species of flowering plant. We can use this attribute to test the performance of the k-means algorithm on the Iris data. Although we understand that using the same data for training and testing may be problematic, for this simple example we will re-use the entire data set.

We can compute the distance of a known data vector $\vec{x}$ to the means $\vec{g}_1$ and $\vec{g}_2$, then compare the classification by minimum distance to the human labeling.

When we do this, we get a perhaps surprising result. There is one data vector, which is labeled as type P, that is classified as type B! This is illustrated in Figure 19.3, where the wayward vector is circled.

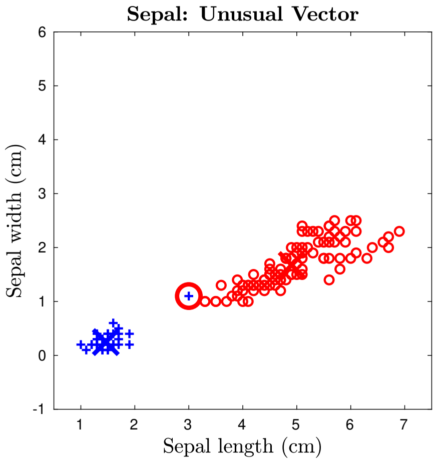

Figure 19.3: Fisher's Iris data set, labeled by a human expert. One data vector is visually in the second cluster, but its distance categorizes it as being in the first cluster. This data vector is circled in the data plot.

We have used a simple concept from linear algebra – the Euclidean distance metric – to automatically cluster this simple data set. In machine learning, other metrics are used and may be described in other computing courses. Our task, in this course, is to better understand how this simple clustering process works and how related algorithms can be used to improve the classification of data.

Extra Notes

This is part of the instructor’s Matlab code for performing k-means clustering on the Iris data set. The built-in Matlab function `kmeans` returns the variable `kmrow` with the means as rows, so this matrix must be transposed into vector form.

```
% Load the data from the Matlab repository; extract the
% sepal length and sepal width as column-oriented data
load fisheriris
x = meas(:,3:4);

% Use the builtin function for K-means clustering
[kmidx, kmrow] = kmeans(x, 2);

% Sort by the first coordinate and transpose into vector form
kmcen = sort(kmrow)’;
```

The variable `kmcen` is a 2×2 matrix, with the first column vector being the centroid of the type-B Iris and the second column vector being the centroid of the type-P Iris.

End of Extra Notes