Clustering

Informatiekunde Reading Group 24/2/2012

Valerio Basile
Outline

- What is clustering
- Hierarchical Clustering
- Point-assignment Clustering
  - K-means algorithms
  - BFR
  - CURE
- GRGPF
- Clustering for Streams
What is Clustering?

- Operation on points that form a space
- Groups the elements closer to each other
- Distance is key
What is Clustering?

Figure 7.1: Heights and weights of dogs taken from three varieties
Space

Euclidean space
• Points are vectors of real numbers
• Natural distance
• Many possible distances (Manhattan, L_inf, …)

Non-euclidean space
• Ad-hoc distances
• Es. strings
Approaches

We can divide (cluster!) clustering algorithms into two groups that follow two fundamentally different strategies.

Hierarchical (or agglomerative)

- Start with each point in its own cluster
- Combine clusters based on “closeness”

Point assignment

- Estimate clusters
- Assign each point to its cluster
The curse of Dimensionality

High-dimensional spaces have a number of unintuitive properties

- Almost all pairs have the same distance
- All angles between vectors are close to 90 degrees
- However, data are usually not random
Hierarchical Clustering

WHILE it is not **time to stop** DO
  pick the best two clusters to merge;
  combine those two clusters into one cluster;
END
Hierarchical Clustering

Decide in advance:

- How will clusters be represented?
- How will we choose which two clusters to merge?
- When will we stop combining clusters?
Hierarchical Clustering

Figure 7.2: Twelve points to be clustered hierarchically
Hierarchical Clustering

Figure 7.3: Combining the first two points into a cluster
Hierarchical Clustering

Figure 7.4: Clustering after two additional steps
Hierarchical Clustering

Figure 7.5: Three more steps of the hierarchical clustering
Hierarchical Clustering

The output can be a number of clusters or the complete tree

Figure 7.6: Tree showing the complete grouping of the points of Fig. 7.2
Hierarchical Clustering

To compute inter-cluster distance we used **centroids**, but there are alternatives.

e.g. minimum of the distances between any two points, one chosen from each cluster
Hierarchical Clustering

- What about Non-Euclidean spaces?
- We can't use centroids, since there is no concept of “middle point”
- Solution: clustroids
Clustering Non-Euclidean spaces

Edit distances between strings

<table>
<thead>
<tr>
<th></th>
<th>ecdab</th>
<th>abecb</th>
<th>aecdb</th>
</tr>
</thead>
<tbody>
<tr>
<td>abcd</td>
<td>5</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>aecdb</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>abecb</td>
<td>4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Clustroid

<table>
<thead>
<tr>
<th>Point</th>
<th>Sum</th>
<th>Max</th>
<th>Sum-Sq</th>
</tr>
</thead>
<tbody>
<tr>
<td>abcd</td>
<td>11</td>
<td>5</td>
<td>43</td>
</tr>
<tr>
<td>aecdb</td>
<td>7</td>
<td>3</td>
<td>17</td>
</tr>
<tr>
<td>abecb</td>
<td>9</td>
<td>4</td>
<td>29</td>
</tr>
<tr>
<td>ecdab</td>
<td>11</td>
<td>5</td>
<td>45</td>
</tr>
</tbody>
</table>
K-means algorithms

- Best known family of point-assignment algorithm
- Assume the number of clusters, $k$, is known in advance
- Also assume Euclidean space
K-means algorithms

Initially choose \( k \) points that are likely to be in different clusters;
Make these points the centroids of their clusters;
FOR each remaining point \( p \) DO
    find the centroid to which \( p \) is closest;
    Add \( p \) to the cluster of that centroid;
    Adjust the centroid of that cluster to account for \( p \);
END;

Figure 7.7: Outline of \( k \)-means algorithms
Picking the right value of $k$

- The number of desired clusters may be known in advance
- Otherwise it can be estimated by looking at clusters diameter (or other measures)

![Graph showing the relationship between average diameter, correct value of $k$, and number of clusters.]

Figure 7.9: Average diameter or another measure of diffuseness rises quickly as soon as the number of clusters falls below the true number present in the data.
The BFR Algorithm

- The Algorithm of Bradley, Fayyad, and Reina is a variant of $k$-means
- Designed for high-dimensional spaces
- Strong assumption on the clusters shape

Figure 7.10: The clusters in data for which the BFR algorithm may be used can have standard deviations that differ along different axes, but the axes of the cluster must align with the axes of the space
The BFR Algorithm

• Initially select $k$ points
• Process data chunks in main memory
• Three sets also in main memory:
  • **Discard** Set: summaries of the clusters
  • **Compressed** Set: summaries of sets of points
  • **Retained** Set: “isolated” points that don't fit into the previous sets
The BFR Algorithm

Figure 7.11: Points in the discard, compressed, and retained sets
The BFR Algorithm

**Summaries** are $2d+1$ values:

- The number of points $N$
- The sum of the components of all the points in each dimension (vector $SUM$)
- The sum of the squares of the components of all the points in each dimension (vector $SUMSQ$)
The BFR Algorithm

Process chunks of data:

• Points that are close to a centroid are added to its cluster

• The other points are clustered along with the retained set. Merge the resulting miniclusters with the compressed set

• Finally, take care of the remaining points and miniclusters
The CURE Algorithm

Clustering Using Representatives is a large-scale, point-assignment clustering algorithm

- Assumes Euclidean space
- No assumptions on clusters shape
- Uses a collection of representative points instead of centroids
The CURE Algorithm

Designed for oddly-shaped clusters

Figure 7.12: Two clusters, one surrounding the other
The CURE Algorithm

Initialization (1)

Figure 7.13: Select representative points from each cluster, as far from one another as possible
The CURE Algorithm

Initialization (2)

Figure 7.14: Moving the representative points 20% of the distance to the cluster’s centroid
The CURE Algorithm

- After initialization, merge clusters with min distance between representative points
- Assign points to clusters based on representative points
The GRGPF algorithm

• Designed for Non-Euclidean spaces
• Mixed approach:
  • Point-assignment
  • Organizes clusters hierarchically
• Let $ROWSUM(p)$ be the sum of the squares of the distances from $p$ to each of the other points in its cluster
The GRGPF algorithm

- Clusters are represented with features:
  - N, the number of points in the cluster
  - The clustroid of the cluster (the point that minimizes ROWSUM)
  - The $k$ points closest to the clustroid
  - The $k$ points furthest from the clustroid
- Clusters are organized in a tree
  - “closer” leaves contain closer clusters
The GRGPF algorithm

- Initialize the tree with a main-memory algorithm
  - Internal nodes hold a sample of the clustroids of the clusters represented by its substree
- For each point, assign it to a cluster by passing it down the tree
  - At each internal node, look at the sample and choose a subtree
  - At a leaf, pick the cluster with the closest clustroid and update the features
The GRGPF algorithm

- Set of closest point used to move clustroids
- Set of furthest points used to merge clusters
- Eventually, clusters are split when they grow too large
Clustering for Streams

- Sliding window of $N$ points
- Query on last $m \leq N$ points
- No assumption on space type
- Clusters change over time
The BDMO Algorithm

- Generalization of DGIM Algorithm
- **Buckets** of points holding size, timestamp, and representations of their clusters
- Every $p$ points
  - create a bucket
  - consider whether to *merge* buckets
- Answer **queries** by merging the buckets that cover the last $m$ points
Clustering in a Parallel Environment

- We use **Map-Reduce**
- In most cases, single Reduce task
- Map tasks:
  - cluster input points
  - return key-value pairs where **key** is always 1 and **value** is the description of a cluster
- Reduce task merge the clusters
Thank you!

Questions?