# Worksheet 09

Name: Shangyuan Chen UID: U58846351

## Topics

- Clustering Review
- Clustering Aggregation

## Clustering Aggregation

| Point | C | P |
|---|---|---|
| A | 0 | a |
| B | 0 | b |
| C | 2 | b |
| D | 1 | c |
| E | 1 | d |

a) Fill in the following table where for each pair of points determine whether C and P agree or disagree on how to cluster that pair.

| Pair | Disagreement |
|---|---|
| A B | 1 |
| A C | 1 |
| A D | 1 |
| A E | 1 |
| B C | 1 |
| B D | 1 |
| B E | 1 |
| C D | 1 |
| C E | 1 |
| D E | 1 |

*1 for disagree

As datasets become very large, this process can become computationally challenging.

b) Given N points, what is the formula for the number of unique pairs of points one can create?

$(N \times (N-1))/2$

Assume that clustering C clusters all points in the same cluster and clustering P clusters points as such:

| Point | P |
|---|---|
| A | 0 |
| B | 0 |
| C | 0 |
| D | 1 |
| E | 1 |
| F | 2 |
| G | 2 |
| H | 2 |
| I | 2 |

c) What is the maximum number of disagreements there could be for a dataset of this size? (use the formula from b)?

**Step-by-Step Calculation:**

- Total number of points, ( N = 9 ) (A, B, C, D, E, F, G, H, I).
- Number of points in each cluster in ( P ):
  - Cluster 0 has 3 points.
  - Cluster 1 has 2 points.
  - Cluster 2 has 4 points.
- **Pairs within each cluster in ( P ) (no disagreements here):**
  - Cluster 0: (\binom{3}{2} = 3) pairs.
  - Cluster 1: (\binom{2}{2} = 1) pair.
  - Cluster 2: (\binom{4}{2} = 6) pairs.
  - Total in-cluster pairs (agreeing): (3 + 1 + 6 = 10).
- **Total possible pairs in ( N ) points:**
  - (\binom{9}{2} = 36) pairs.
- **Pairs from different clusters (disagreements):**
  - Total pairs - In-cluster pairs = ( 36 - 10 = 26 ) disagreements.

Thus, there are 26 disagreements between clustering ( C ) and clustering ( P ) when considering the maximum possible disagreements given ( C ) clusters all in one group and ( P ) as specified.

d) If we look at cluster 0. There are (3 x 2) / 2 = 3 pairs that agree with C (since all points in C are in the same cluster). For each cluster, determine how many agreements there are. How many total agreements are there? How many disagreements does that mean there are between C and P?

To find the number of agreements and disagreements between the clusterings ( C ) and ( P ), we need to calculate the number of pairs that are both in the same cluster in ( P ) (as they are all in the same cluster in ( C ) due to ( C ) clustering all points together).

**Clusters in ( P ):**

- Cluster 0: 3 points
- Cluster 1: 2 points
- Cluster 2: 4 points

**Agreements in Each Cluster:** These are the pairs of points that are together in each cluster in ( P ) and would also be together in ( C ).

- **Cluster 0 (3 points):** [ \binom{3}{2} = {3 * 2}/{2} = 3 agreements ]
- **Cluster 1 (2 points):** [ \binom{2}{2} = {2 * 1}/{2} = 1 agreement ]
- **Cluster 2 (4 points):** [ \binom{4}{2} = {4 * 3}/{2} = 6 agreements ]

**Total Agreements:** [ 3 + 1 + 6 = 10 agreements ]

**Total Possible Pairs in the Dataset (All 9 points):** [ \binom{9}{2} = {9 * 8}{2} = 36 total possible pairs ]

**Total Disagreements:** Given all points are in a single cluster in ( C ), any pair that is in different clusters in ( P ) is a disagreement. The total disagreements can thus be calculated as the difference between the total possible pairs and the total agreements.

[ 36 - 10 = 26 disagreements ]

So, there are 10 agreements and 26 disagreements between ( C ) and ( P ) when considering how they cluster the points.

e) Assuming that filtering the dataset by cluster number is a computationally easy operation, describe an algorithm inspired by the above process that can efficiently compute disagreement distances on large datasets.

## Algorithm Description

1. **Input Preparation**:
   - Input the dataset ( D ) and the two clusterings ( C ) and ( P ).
   - Ensure each point in the dataset has an identifier to make pairwise comparisons feasible.
2. **Cluster Extraction**:
   - For both clusterings ( C ) and ( P ), create a mapping from each cluster label to the list of points assigned to that label. This can be represented as dictionaries or hash

tables, where keys are cluster labels and values are lists of points.

3. **Agreement Calculation**:
    - For each cluster in ( C ), do the following:
        - For each cluster in ( P ), determine the intersection of points between the current cluster in ( C ) and the current cluster in ( P ). This gives you the points that both clusterings agree are in the same group.
        - Calculate the number of pairs in each intersection using the combination formula ( \binom{n}{2} ), where ( n ) is the number of points in the intersection.
        - Sum these values to get the total number of agreeing pairs for the clusters considered.
4. **Total Pairs Calculation**:
    - Calculate the total number of pairs in the dataset using ( \binom{N}{2} ), where ( N ) is the total number of points in the dataset.
5. **Disagreement Calculation**:
    - Subtract the total number of agreeing pairs (from step 3) from the total number of pairs (from step 4) to get the number of disagreeing pairs.
6. **Optimization**:
    - To optimize the computation of intersections in step 3, utilize efficient data structures such as sets in programming languages that support fast set operations.
    - Parallel processing can be used when processing different clusters in ( C ) and ( P ) to expedite the agreement calculations.
7. **Output**:
    - Return or print the total number of disagreements.

## Considerations for Large Datasets

- **Memory Efficiency**: Storing points in clusters using sets or hash tables helps in managing memory by avoiding duplication and facilitating fast lookups.
- **Parallel Processing**: Since the computation for each cluster pair is independent of others, this process is highly parallelizable, allowing for significant speed-ups on multi-core systems or distributed computing environments.
- **Incremental Updating**: If the dataset or cluster assignments change incrementally (few points change clusters), the algorithm can be adapted to update the disagreement count dynamically rather than recomputing from scratch.

This algorithm leverages the structure of cluster assignments to efficiently determine agreements and disagreements, crucial for handling large datasets where direct pairwise comparisons would be computationally prohibitive.