UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

OTDM - Project 3 - Clustering

Authors: Julian Fransen, Danila Kokin

Date: 2025-01-06

Contents

Description of Data Matrix A

Title: Wine Data (Adapted for Unsupervised Learning)

Sources:

- **Original Source**: UCI Machine Learning Repository
- **Original Dataset**: Wine Dataset
- **Creator**: Forina et al. (1991), Analytical Chemistry

Relevant Information:

These data are the results of a chemical analysis of wines grown in the same region in Italy, derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. The dataset is adapted for unsupervised learning by removing the labels that classify the types of wines.

- **Data Usage**: Frequently used for clustering, PCA, and classification studies.
- **Domain**: Chemistry, wine analysis, and machine learning.

Number of Instances:

- 178 wine samples.

Number of Attributes:

- 13 numeric continuous attributes.

Attribute Information:

1. Alcohol
2. Malic acid
3. Ash
4. Alcalinity of ash
5. Magnesium
6. Total phenols
7. Flavanoids
8. Nonflavanoid phenols
9. Proanthocyanins
10. Color intensity
11. Hue
12. OD280/OD315 of diluted wines
13. Proline

Missing Attribute Values:

- None.

Preprocessing:

The dataset was scaled using Z-score normalization to standardize the features to zero mean and unit variance for better interpretability and clustering performance. After scaling, the Euclidean distance was calculated, and this matrix of distances was formatted to AMPL compatible .dat files.

Summary Statistics:

Feature	Min	Max	Mean	Standard Deviation
Alcohol	11.03	14.83	13.00	0.81
Malic acid	0.74	5.80	2.34	1.12
Ash	1.36	3.23	2.36	0.27
Alcalinity of ash	10.60	30.00	19.49	3.34
Magnesium	70.00	162.00	99.74	14.28
Total phenols	0.98	3.88	2.29	0.63
Flavanoids	0.34	5.08	2.03	0.99
Nonflavanoid phenols	0.13	0.66	0.36	0.12
Proanthocyanins	0.41	3.58	1.59	0.57
Color intensity	1.28	13.00	4.71	2.43
Hue	0.48	1.71	0.96	0.23
OD280/OD315 of diluted wines	1.27	4.00	2.61	0.71
Proline	278.00	1680.00	746.89	314.91

The AMPL `.mod` and `.dat` Files

Sample of `wine.dat` File:

```
param m := 178;
param k := 3;
param d: 1 2 ... 177 178 :=
1   0.000000  3.497535 ...  6.078781  7.184421
2   3.497535  0.000000 ...  6.094927  7.367719
...
178 7.184421  7.367719 ...  3.324276  0.000000;
```

Contents of `clustering.mod` file:

```
/#Below we define the AMPL model for clustering
param m;
param k;
param d{1..m, 1..m};
var x{1..m, 1..m} binary;

minimize f: sum{i in 1..m, j in 1..m} d[i,j]*x[i,j];
subject to c1{i in 1..m}: sum{j in 1..m} x[i,j] = 1;
/# This constraint ensures every point belongs to 1 cluster
subject to c2: sum{j in 1..m} x[j,j] = k;
/# This constraint ensure that there are exactly k clusters
subject to c3{i in 1..m, j in 1..m}: x[j,j] >= x[i,j];
/# This constraint ensures that a point can only be part of a cluster that exists
```

The optimal solution obtained with AMPL.

Below you can find the optimal solution of the Wine data found by AMPL

```
## CPLEX 22.1.1.0: timing 1
##
## Times (seconds):
## Input =  0.056011
## Solve =  13.7025
## Output = 0.053996
```

```
## CPLEX 22.1.1.0: optimal integer solution; objective 500.929194
## 5298 MIP simplex iterations
## 0 branch-and-bound nodes
##      Non-zero clusters
## Cluster 36 has 74 points
## Cluster 107 has 55 points
## Cluster 149 has 49 points
```

Below you can find the optimal solution of the well known Iris data found by AMPL

```
## CPLEX 22.1.1.0: timing 1
##
## Times (seconds):
## Input =  0.038188
## Solve =  24.5653
## Output = 0.038111
## CPLEX 22.1.1.0: optimal integer solution; objective 131.270316
## 4860 MIP simplex iterations
## 0 branch-and-bound nodes
##      Non-zero clusters
## Cluster 8 has 50 points
## Cluster 95 has 44 points
## Cluster 148 has 56 points
```

MST: Background

The Minimum Spanning Tree (MST) computation and subsequent clustering process were implemented using Python in a Jupyter Notebook environment (.ipynb file). The following steps describe the methodology and tools used:

Preprocessing

- Outlier Removal: outlier detection and removal function based on the Interquartile Range (IQR) rule was implemented. Data points beyond 1.5 times the IQR from the first quartile (Q1) or third quartile (Q3) were identified as outliers and removed. This ensured the data was clean before clustering.
- Data Normalization: cleaned dataset was standardized using z-score normalization, where each feature was transformed to have a mean of 0 and a standard deviation of 1.

MST Construction

Software and Libraries

- The MST computation was implemented using the Python library NetworkX.
- Pairwise distances between data points were calculated using SciPy's pdist function, which computes the Euclidean distance.
- The distance matrix was converted into a weighted graph using NetworkX's Graph class.

Algorithm Choice

- Two MST algorithms, Prim's and Kruskal's, were available, and the user could specify which to use by setting the algorithm parameter to 'prim' or 'kruskal'. This flexibility was provided by NetworkX's minimum_spanning_tree function, which supports both algorithms.

Steps

- A distance matrix was computed using pdist.
- A weighted graph was constructed where nodes represented data points and edges represented pairwise distances.
- The MST was computed using the specified algorithm.

Clustering by MST Partitioning

- Edge Removal: to partition the MST into the desired number of clusters (k), the `k-1` largest edges (based on their weights) were removed. This step disconnected the MST into k connected components, each representing a cluster.
- Connected Components: the connected components of the resulting graph were identified using NetworkX's connected_components function.
- Cluster Assignments: each connected component was assigned a unique cluster label, and the data points in each component were grouped accordingly.

The heuristic solution obtained with the minimum spanning tree procedure.

Wine Dataset

In the following analysis, we applied Prim's and Kruskal's algorithms to create Minimum Spanning Trees (MSTs) on the wine dataset. The clustering results are visualized in both 2D and 3D plots:

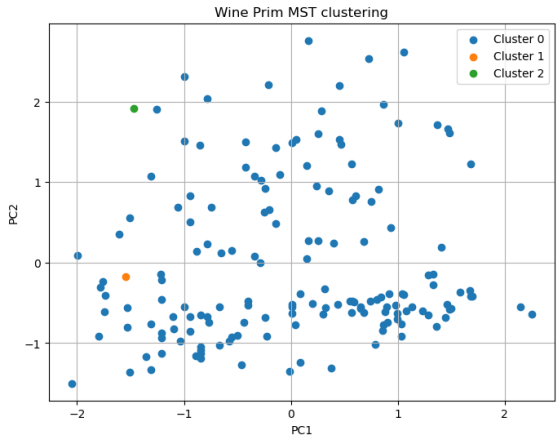

Figure 1: Prim's algorithm on wine dataset

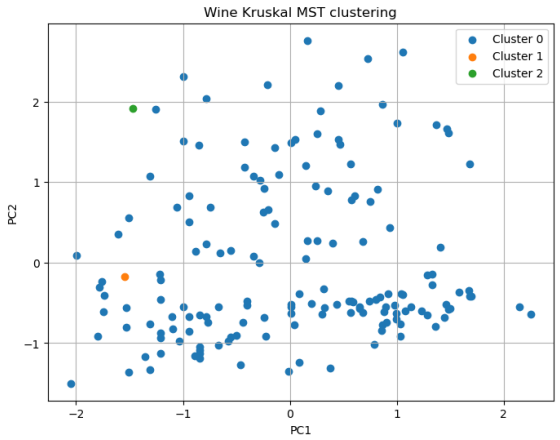

Figure 2: Kruskal's algorithm on wine dataset

Initially, the clustering results appeared suboptimal, with clusters formed as follows:

- 1 obseravation in cluster 2
- 3 obseravation in cluster 1
- 174 observstions (the rest) in cluster 0

This distribution raised concerns about the algorithm's performance. However, it is important to note that the variance explained by the first three principal components is relatively low:

```
Explained variance by each principal component: [0.36198848 0.1920749  0.11123631]
```

Given this, the clustering results may be valid but are challenging to interpret visually in 2D or 3D projections. To further evaluate the algorithm's performance, we tested it on a dataset with lower dimensionality.

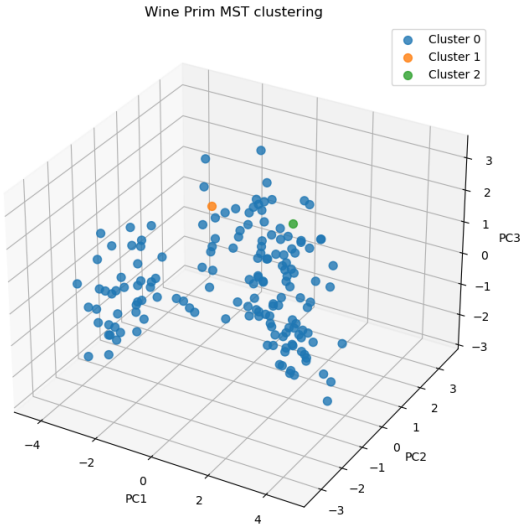

Figure 3: Prim's algorithm on wine dataset

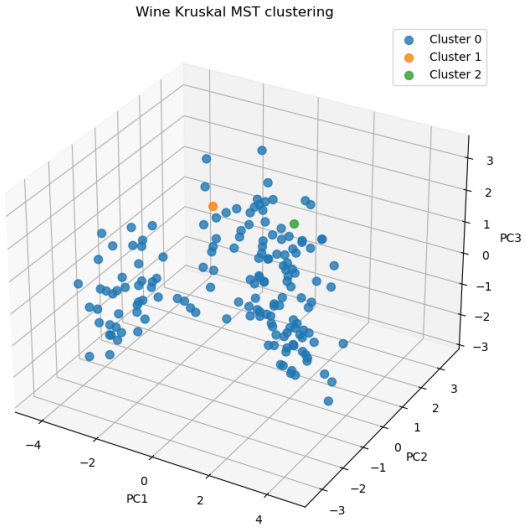

Figure 4: Kruskal's algorithm on wine dataset

Iris Dataset

The Iris dataset, a widely known benchmark, includes only four features and three distinct plant classes. After applying outlier detection, scaling, and cleaning, we ran the same MST algorithms. The resulting clusters are illustrated below:

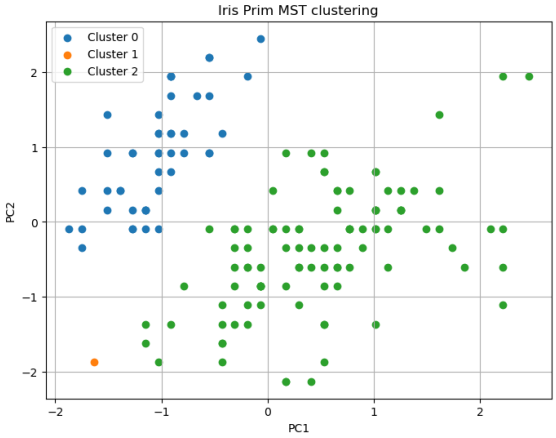

Figure 5: Prim's algorithm on iris dataset

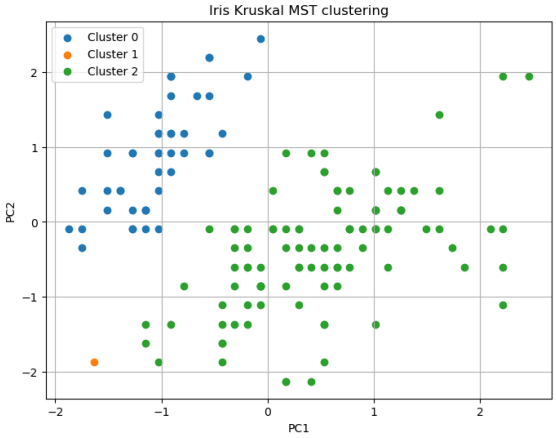

Figure 6: Kruskal's algorithm on iris dataset

The clustersdistribution is:

- 100 obseravation in cluster 2
- 1 obseravation in cluster 1
- 49 observstions in cluster 0

In this case, the variance explained by the first three principal components is significantly higher, with a cumulative total nearing 90%:

```
Explained variance by each principal component: [0.72770452 0.23030523 0.03683832]
```

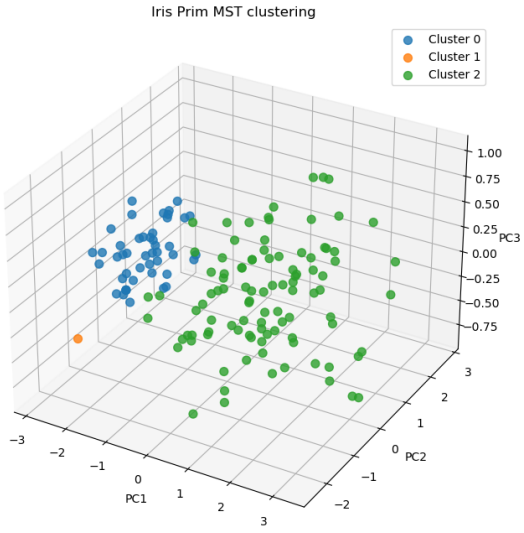

Figure 7: Prim's algorithm on iris

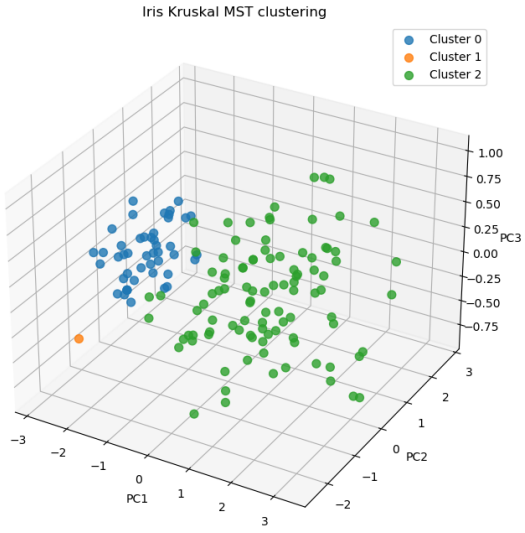

Figure 8: Kruskal's algorithm on iris

This higher explained variance allows for a more accurate and visually interpretable clustering. The 3D plots particularly highlight well-defined clusters, demonstrating that the algorithms perform well.

Comparison of results

The clustering results obtained from the AMPL optimization model and the MST approach show the fundamental trade-offs between accuracy and efficiency in solving optimization problems. The AMPL solution produced an objective value of **500.93** on the wine dataset, reflecting a highly compact clustering arrangement with minimal within-cluster distances. In contrast, the heuristic approach resulted in a significantly higher total within-cluster distance of **72,876.83**, indicating a less efficient clustering outcome. Furthermore, the number of data points per cluster for the wine data are **74, 55, and 49** for AMPL, compared to **1, 3, and 174** for MST. For the Iris dataset, they are **50, 44, and 56** for AMPL, compared to **100, 49, and 1** for MST.

The disparity in objective function values underscores the strengths and limitations of each method. The AMPL model excels in achieving global optimization and produces obviously superior clustering quality. However, this approach requires substantial computational resources and can be too time-intensive, particularly for larger datasets. On the other hand, the heuristic MST-based approach offers a more scalable alternative, even though the clustering quality is often compromised, as seen in its sub-optimal performance on the wine dataset.

Despite these challenges, the experiments on the Iris dataset revealed the true potential of the AMPL model when applied to datasets with clearer structures and fewer dimensions. The AMPL solution achieved an exceptional objective function value of **131.27**, significantly outperforming the MST heuristic, which produced a value near **9,636.68**. These results emphasize that the AMPL model is particularly well-suited for datasets where the underlying clusters are more distinct and interpretable. The MST heuristic, while efficient, struggles in such cases to achieve meaningful cluster separations, as highlighted by the extreme imbalance in cluster sizes for both datasets. This comparison underscores the importance of selecting a method tailored to the dataset's characteristics and the goals of the analysis.

Conclusion

The assignment highlighted several challenges and observations in implementing and comparing clustering methods. The AMPL optimization model consistently delivered high-quality results, but its computational

demands are significant. Conversely, the heuristic MST approach are supposed to excell in speed and scalability, but in our experience their results were very poor, as is evident by its high objective function values.