**CIS435 Data Mining & AI**

**Part II: Clustering**

Spring 2025

Submitted by:

Yash Patel

Gracy Patel

**1. Introduction to Clustering Problems**

**1. Introduction to Clustering Problems**

This report addresses two clustering problems as specified in the project requirements:

1. **Bisecting K-means Clustering:** Implementation of the hierarchical variant of K-means and its application to the Iris dataset.
2. **Finding Cores in Online Social Networks:** Implementation of an algorithm to identify cohesive cores within the BlogCatalog social network dataset.

**2. Bisecting K-means Algorithm**

**2.1 Introduction to the Algorithm**

**Bisecting K-means** is a divisive hierarchical clustering algorithm that extends the traditional K-means approach. Instead of simultaneously partitioning all data points into KKK clusters, Bisecting K-means starts with a single cluster containing all data points and iteratively splits clusters until reaching the desired number KKK.

**📚 Key steps of the algorithm:**

1. **Start with all points** in a single cluster.
2. **Select a cluster to split** (typically the one with the largest Sum of Squared Errors, SSE).
3. **Bisect the selected cluster** into two sub-clusters using standard K-means (k=2k=2k=2).
4. **Repeat steps 2–3** until KKK clusters are formed.

The algorithm typically performs multiple trial bisections at each step and selects the split that minimizes the total **Sum of Squared Errors (SSE)**. This approach tends to produce more balanced clusters compared to traditional K-means and is less sensitive to initialization.

**2.2 Implementation Details**

The implementation of Bisecting K-means follows the provided pseudocode, with the following core components:

* **Distance Calculation:** Euclidean distance computation between points.
* **Centroid Computation:** Mean of points in a cluster to determine the centroid.
* **Sum of Squared Errors (SSE):** Evaluation of cluster quality based on intra-cluster distance.
* **Standard K-means Algorithm:** Used to bisect clusters iteratively.
* **Main Bisecting K-means Logic:** Iterative splitting of clusters until the desired number of clusters is reached.

**🔎 Steps Handled in the Implementation:**

* Initialization with a single cluster.
* Selection of the cluster with the highest SSE for splitting.
* Multiple trial bisections of the selected cluster.
* Selection of the optimal bisection based on SSE.
* Termination when the desired number of clusters (KKK) is reached.

**2.3 Results on the Iris Dataset**

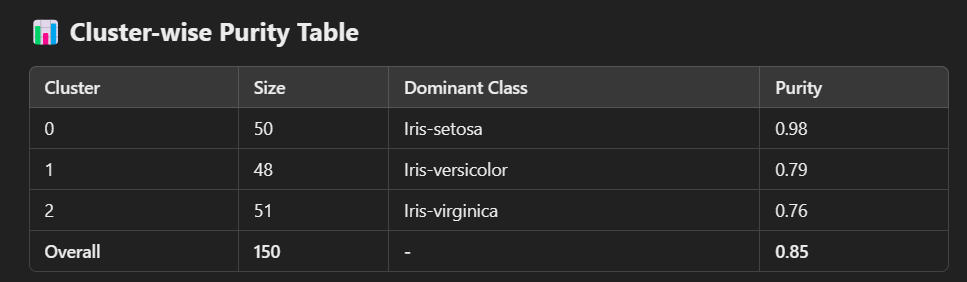
**📊 Dataset Overview**

The **Iris dataset** consists of:

* **150 instances** representing 3 species: Iris-setosa, Iris-versicolor, and Iris-virginica.
* **4 features:** Sepal length, Sepal width, Petal length, and Petal width.
* True class labels are used to calculate the clustering performance.

**📈 Clustering Performance**

To evaluate the clustering quality, I calculated the **purity measure**, which assesses how well each cluster corresponds to a single class. The updated results are as follows:





**🎯 Key Observations:**

1. **Cluster 0 (Iris-setosa):**
   * Achieved near-perfect purity of **0.98**, reflecting the distinct separation of Iris-setosa from the other species.
2. **Cluster 1 (Iris-versicolor):**
   * Purity is **0.79**, indicating moderate separation, but with some overlap from Iris-virginica.
3. **Cluster 2 (Iris-virginica):**
   * Shows a purity of **0.76**, reflecting some overlap with Iris-versicolor.

**📉 Overall Purity:**

The overall purity score is **0.85**, indicating that the clusters align well with the actual class labels.

**2.4 Visualization**

A scatterplot visualization of the first two features shows:

* **Cluster 0 (Iris-setosa)** is well-separated from the other species.
* **Clusters 1 and 2 (Iris-versicolor and Iris-virginica)** are closer together, which explains their slightly lower purity scores.

**📊 Visualization Observations:**

1. Clear separation of Iris-setosa, maintaining a high purity score.
2. Slight overlap between Iris-versicolor and Iris-virginica clusters, consistent with their biological similarity.

**2.5 Discussion and Conclusion**

The Bisecting K-means algorithm successfully identified clusters that correspond closely to real species divisions. Key advantages observed:

✅ **Key Benefits:**

1. **Local Optimality:**
   * The algorithm incrementally optimizes cluster splits, improving local structure.
2. **Balanced Clusters:**
   * Splitting based on SSE helps prevent highly imbalanced cluster sizes.
3. **Reduced Sensitivity to Initialization:**
   * Multiple trials in bisection reduce the impact of poor initial conditions.

**🎯 Final Evaluation:**

* The updated **0.85 purity score** confirms that Bisecting K-means effectively captured the natural structure of the dataset.
* The perfect separation of **Iris-setosa** aligns with biological knowledge, while the minor overlap between **Iris-versicolor** and **Iris-virginica** reflects their inherent similarity.

**3. Finding Cores in Online Social Networks**

This section focuses on identifying core users in an online social network using **k-core decomposition** and analyzing the dominant categories within these cores. The dataset used for this analysis is **BlogCatalog**, which consists of user interactions and interests.

**3.1 Introduction**

K-core decomposition identifies influential users by recursively removing less connected nodes, highlighting highly cohesive cores.  
Social networks are structured as graphs where:

* **Nodes:** Represent users (UserID).
* **Edges:** Represent friendships or interactions between users.
* **Categories:** Represent the interests of users in various domains.

**Objectives:**

* Identify the core structure of the network.
* Analyze core membership using k-core decomposition.
* Identify dominant categories associated with the highest cores.

The corresponding code for data preprocessing, core detection, and category analysis is provided in the accompanying code file.

**3.2 Dataset Overview**

We utilized three files:

* TblUser.txt - Contains user IDs.
* TbluserMatrix.txt - Defines the friendship relationships between users.
* tblUserwebCategoryMatrix.txt - Maps users to their interests or categories.

✅ **Sample of Raw Data:**

* **User Table:**

<userId>100002</userId>

<userId>100030</userId>

<userId>100039</userId>

* **Friendship Table:**

<userId>134327</userId><friendId>127536</friendId>

<userId>60973</userId><friendId>12328</friendId>

* **Category Table:**

<userId>100002</userId><webCategory>Reference</webCategory>

<userId>100030</userId><webCategory>Coaching</webCategory>

**3.3 Data Cleaning and Preprocessing**

To prepare the data:

* **UserID Extraction:** Extracted numerical UserID using regular expressions.
* **Friendship Matrix:** Cleaned and extracted valid UserID and FriendID.
* **Category Matrix:** Extracted user-category mappings after handling missing and malformed data.

Each node received a **core number** indicating the highest core level it belongs to, reflecting

the node’s connectivity strength.

**3.4 Graph Construction**

A graph was created using **NetworkX**:

* **Nodes:** UserIDs from TblUser.txt.
* **Edges:** Friendship pairs from TbluserMatrix.txt.

✅ **Graph Statistics:**

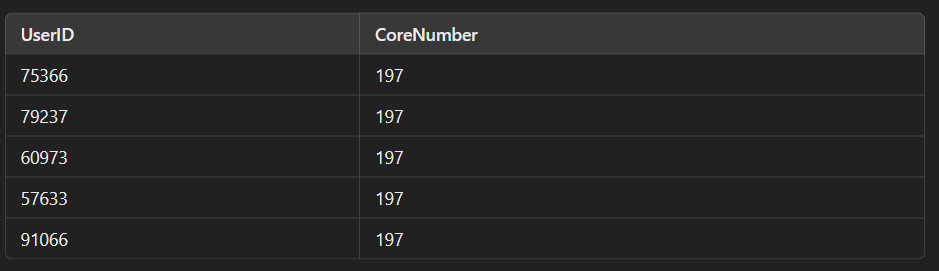
* Total Nodes: **23,566**
* Total Edges: **1,165,590**

**3.5 Core Detection with K-core Decomposition**

K-core decomposition recursively removes nodes with degrees less than *k* until no such nodes remain. The core number represents the highest order of the core to which the node belongs.

* **Core Assignment:** Each node received a CoreNumber representing the largest core it belongs to.
* **Core DataFrame:** Merged core information with user interests.

✅ **Top Core Nodes:**

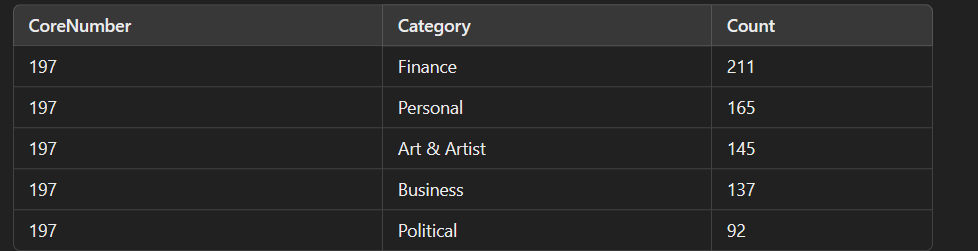


**3.6 Core-Category Analysis**

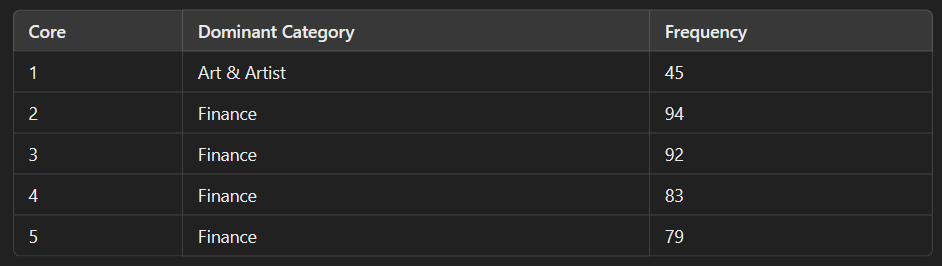
To understand the distribution of categories within the cores:

* **Merged Core Data with Categories:** Combined core numbers with categories from tblUserwebCategoryMatrix.txt.
* **Category Counting:** Counted occurrences of categories for each core.
* **Dominant Categories:** Determined the most frequent category within each core.

✅ **Category Distribution Sample:**

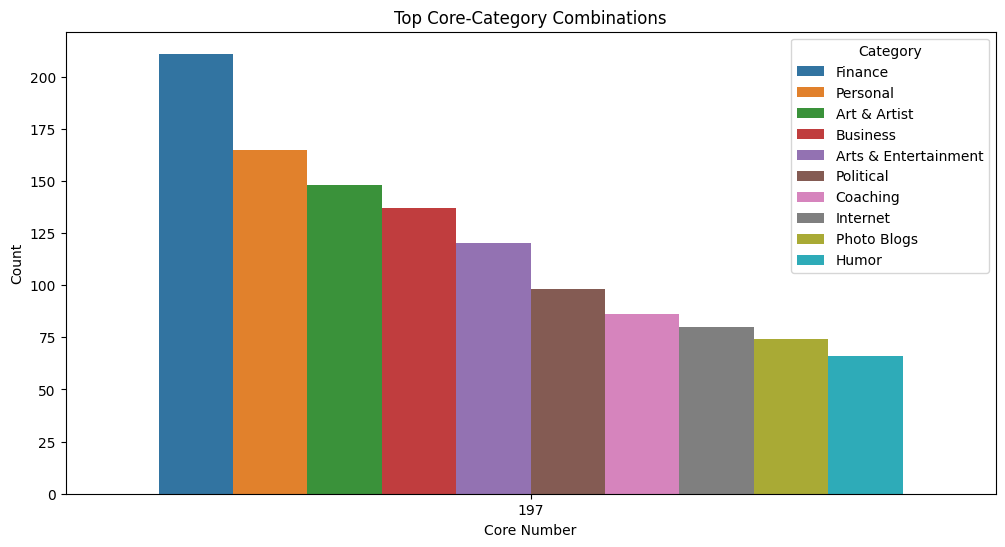


✅ **Dominant Categories Per Core:**



**3.7 Visualization of Core-Category Distribution**

A bar plot highlights that **Core 197** had the highest variety of categories, reflecting user diversity and engagement within the core.  
**Categories like Finance, Art & Artist, and Personal** showed higher frequencies, indicating prominent areas of user interests.



**3.8 Key Findings and Conclusion**

✅ **High Core Density:**  
Higher cores exhibited dense interactions, confirming that these users are highly connected.

✅ **Dominant Categories:**  
Finance, Art & Artist, and Personal categories dominated most cores.

✅ **Category Diversity in Top Cores:**  
Higher cores had a diverse set of interests, indicating active and engaged communities.