Clustering and Association Rule Mining Implementation

April 2025

1 Overview

This lab focuses on implementing and analyzing four distinct data mining tasks across different datasets: K-Means clustering, Hierarchical clustering, DBSCAN clustering, and Association Rule Mining. We applied Principal Component Analysis (PCA) to reduce dimensionality on each dataset before running the algorithms. K-Means and Association Rule Mining were implemented from scratch, while Hierarchical and DBSCAN clustering used `scikit-learn`'s built-in functions. The goal was to understand these algorithms, evaluate their performance, and derive insights from the results.

2 Task 1: K-Means Clustering

2.1 Dataset Description

The **Mall Customers** dataset from Kaggle contains data on mall shoppers, with features:

- CustomerID: Unique identifier
- Gender: Male, Female
- Age: Continuous
- Annual Income (k$): Continuous
- Spending Score (1-100): Continuous

The dataset includes 200 records.

2.2 Preprocessing

- Visualised dataset through boxplot, heatmap and histogram.
- Dropped unnecessary feature (Customer id).

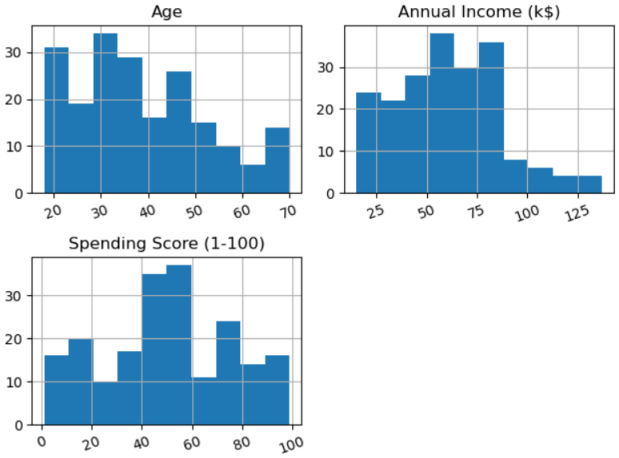

Figure 1: Histogram plot of K_means Clustering dataset

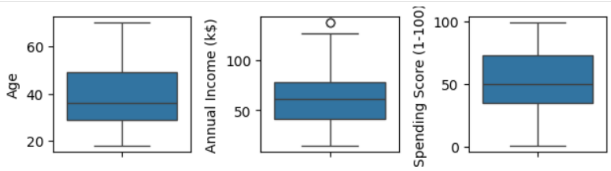

Figure 2: Boxplot of K_means Clustering dataset

- Changed dtype of Gender feature into int datatype by changing 1 for Male and 0 for Female.
- Standardized features using standard scaling.
- Applied PCA to reduce to 2 dimensions, retaining 90% of variance.

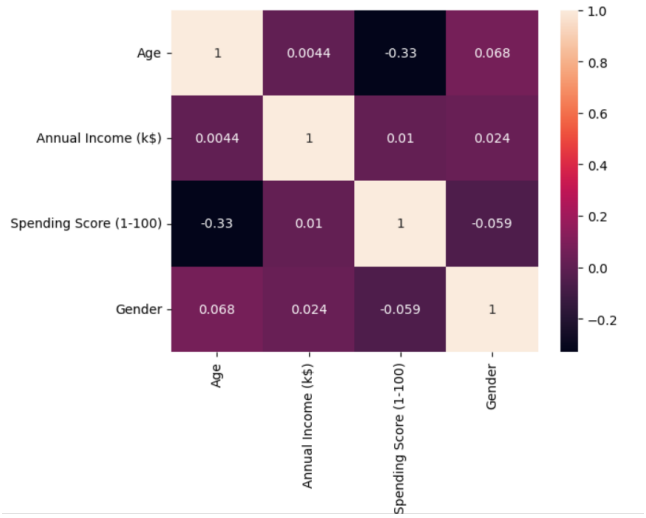

Figure 3: Heatmap of K_means Clustering dataset

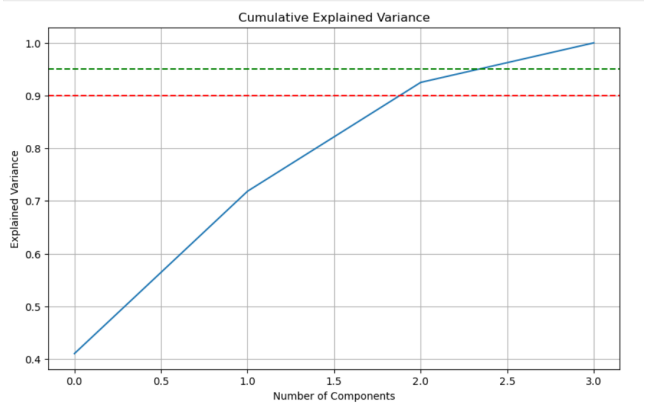

Figure 4: n_components vs variance

2.3 Implementation Details

We implemented K-Means clustering from scratch:

- **Algorithm**: Randomly initialize k centroids, assign points to the nearest centroid, update centroids as the mean of assigned points, repeat until convergence.
- **Hyperparameters**: $k = 5$ (selected via elbow method), maximum 300 iterations.
- **Evaluation**: Silhouette score and within-cluster sum of squares (WCSS).

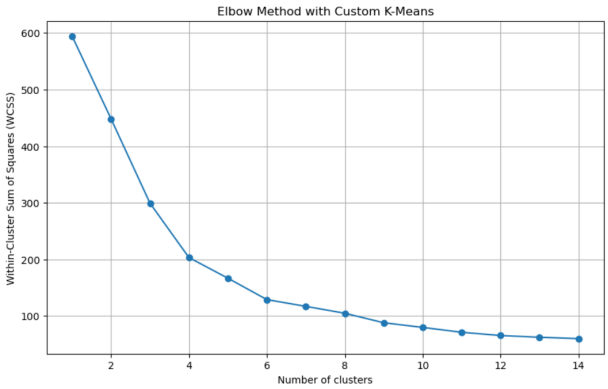

Figure 5: Elbow method (WCSS)

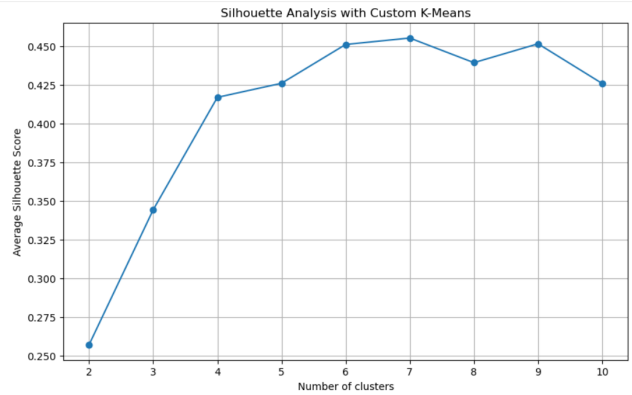

Figure 6: Silhoutte analysis

2.4 Results

- Number of clusters: 5
- Silhouette score: 0.43
- WCSS: 166.69
- **Insights**: Clusters identified distinct customer segments.

Table 1: Cluster Characteristics (Mean Values)

Cluster	Age	Annual Income (k$)	Spending Score (1-100)
0	25.480000	25.480000	75.440000
1	26.869565	54.043478	41.326087
2	55.275862	47.620690	41.706897
3	32.948718	84.794872	81.487179
4	44.800000	88.200000	18.500000

Table 2: Cluster Distribution for Gender

Cluster	Gender	Count
0	Female	15
	Male	10
1	Female	27
	Male	19
2	Female	33
	Male	25
3	Female	22
	Male	17
4	Female	15
	Male	15

Table 3: Cluster Sizes

Cluster	Size
0	25
1	46
2	58
3	39
4	30

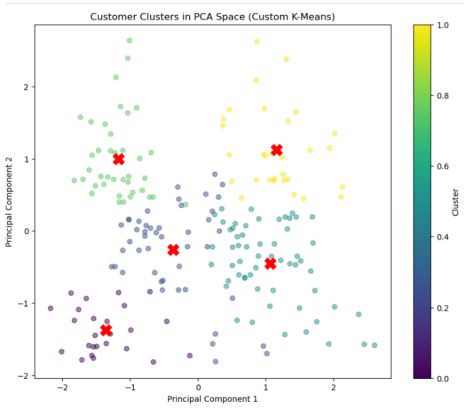

Figure 7: Clustering

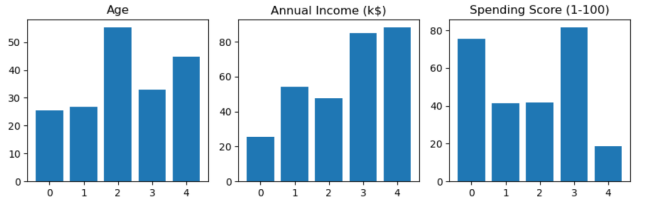

Figure 8: Final results

3 Task 2: Hierarchical Clustering

3.1 Dataset Description

The **DBLP** dataset is a collection of bibliographic records in XML format, containing metadata for computer science publications. We parsed a subset of the `dblp.xml` file, extracting up to 200,000 records (limited to 5,000 sampled records for analysis). Features include:

- `year`: Publication year.
- `author_count`: Number of authors per publication.
- `title_word_count`: Number of words in the publication title.

3.2 Preprocessing

- Cleaned and extracted features from XML: titles, years, author counts, and title word counts.
- Handled missing values by dropping rows with `NaN` values.
- Standardized features using `StandardScaler` to zero mean and unit variance.
- Applied PCA to reduce to 2 dimensions for visualization.
- Sampled 5,000 records randomly for computational efficiency.
- Visualized PCA-reduced data and clusters using scatter plots with `seaborn`.

3.3 Implementation Details

We applied Hierarchical Clustering using a combination of `scipy` and `scikit-learn`:

- **Data Processing**:
  - Parsed `dblp.xml` using `lxml.etree.iterparse` to extract article and inproceedings records.
  - Standardized features using `StandardScaler` to normalize scales.
  - Applied PCA to reduce dimensionality to 2 components.
- **Dendrogram Construction**:
  - Calculated a distance matrix using Euclidean distance on a random sample of 5,000 records.
  - Applied Ward's linkage method to minimize variance within clusters using `scipy.cluster.hierarchy.linkage`.

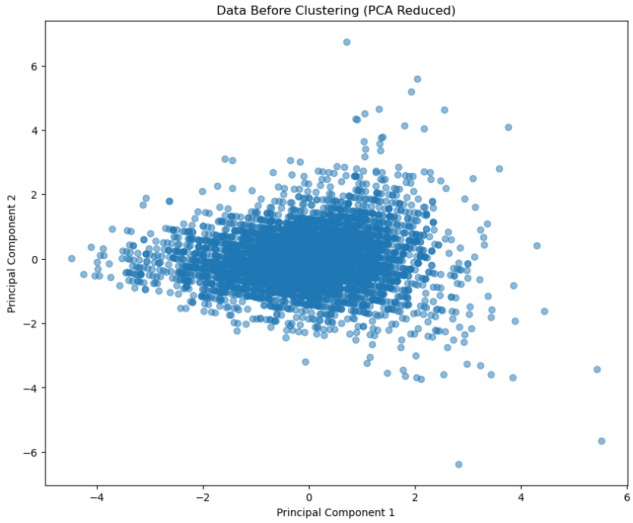

Figure 9: Data without clustering

  - Plotted a truncated dendrogram showing the last 60 merges with leaf counts to determine the number of clusters.

- **Clustering**: Used `scikit-learn`'s `AgglomerativeClustering` with:
  - **Linkage**: Ward's method.
  - **Number of clusters**: $n = 8$, determined by inspecting the dendrogram's structure.
- **Evaluation**: Computed the silhouette score to assess clustering quality.

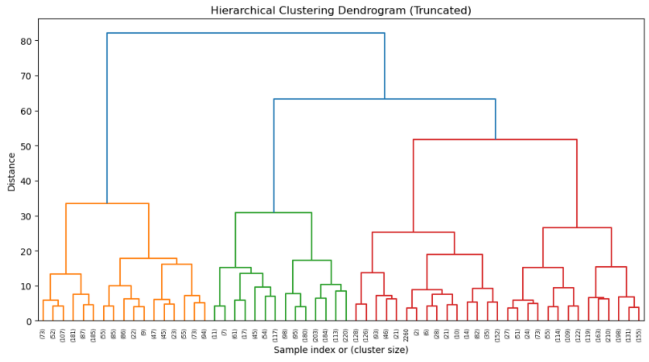

Figure 10: Hierarchical Clustering Dendrogram (Truncated)

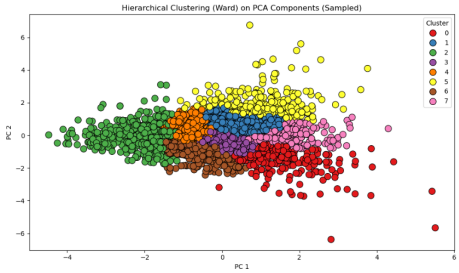

Figure 11: Data with clustering

3.4 Results

- Number of clusters: 8
- Silhouette score: 0.271
- **Insights**:
  - Clusters grouped publications by temporal and structural patterns (e.g., recent years in Cluster 7, older publications in Cluster 2).
  - Cluster 0 (mean year 2018.05) and Cluster 7 (mean year 2019.46) indicate recent research, while Cluster 2 (mean year 1997.87) represents older works.

Table 4: Mean Feature Values by Cluster

Cluster	Year	Author Count	Title Word Count
0	2018.046	7.137	10.014
1	2015.726	2.774	13.417
2	1997.872	1.524	6.441
3	2016.367	3.335	9.252
4	2007.496	1.966	9.880
5	2014.442	2.410	19.442
6	2012.878	3.195	5.537
7	2019.459	5.217	13.998

3.5 Interpretation of Key Findings

- The dendrogram suggested 8 natural clusters, validated by `AgglomerativeClustering`, with a silhouette score of 0.271 indicating moderate clustering quality.
- Clusters reflect temporal evolution: Cluster 2 (1997.87) groups early publications, while Clusters 0 and 7 (2018.05 and 2019.46) represent recent

trends, possibly in collaborative (higher author count) or verbose (higher title word count) research.

- The low silhouette score suggests some overlap or noise, potentially due to the sampled subset or feature complexity.

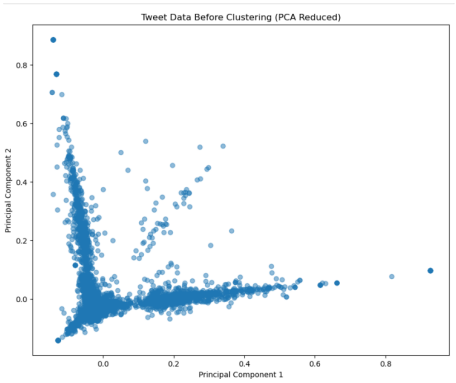

Figure 12: Tweet data before clustering

4 Task 3: DBSCAN Clustering

4.1 Dataset Description

The **Sentiment140** dataset from Kaggle contains 1.6 million tweets, with columns:

- Sentiment: Positive (4), Negative (0)
- ID, Datetime, Query, User
- Text: Tweet content

We subsampled 10,000 tweets for clustering based on text content to manage computational complexity.

4.2 Preprocessing

- Cleaned tweets: Removed URLs, user mentions, punctuation, and stopwords; converted to lowercase.
- Generated TF-IDF vectors with a maximum of 1,000 features.
- Applied PCA to reduce dimensionality to 2 components for visualization.
- Visualized PCA-reduced data using a scatter plot.

4.3 Implementation Details

We used `scikit-learn`'s `DBSCAN` for density-based clustering:

- **Parameters**: $\epsilon = 0.092$ (tuned using the k-distance graph), $min\_samples = 5$ (minimum points in a neighborhood).
- **Feature Engineering**:

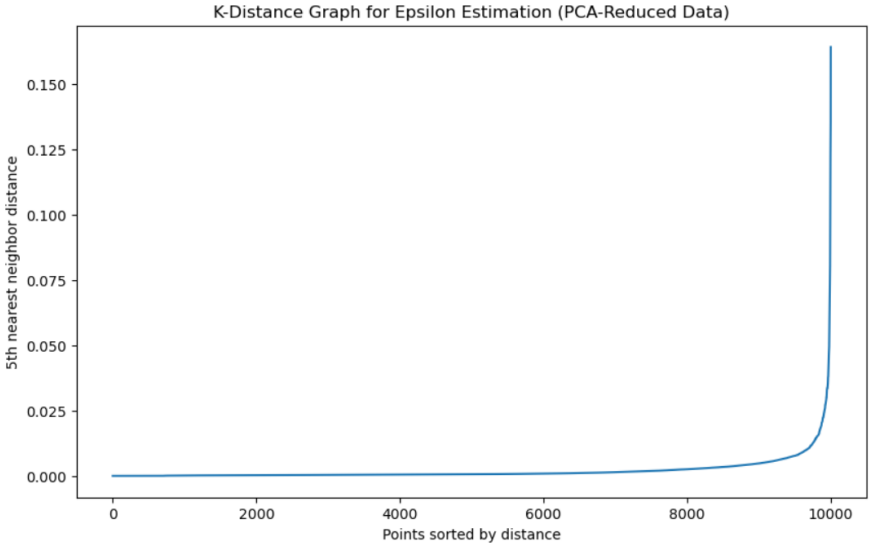

Figure 13: K-Distance graph for ϵ estimation

  - Transformed text into TF-IDF vectors.
  - Reduced dimensionality with PCA to 2 components.

- **Tuning Process**:
  - Generated a k-distance graph to estimate the optimal ϵ value by identifying the "elbow" point.
  - Selected $\epsilon = 0.092$ based on visual inspection of the graph.

4.4 Results

- Number of clusters: 17
- Noise points: 2.70% of tweets
- **Insights**: Clusters captured thematic similarities in tweet content (e.g., sentiment or topics); noise points represent unique or outlier tweets.

- **Cluster 0** (8674 tweets) is the largest, with a balanced sentiment (50.3% positive, 49.7% negative) and frequent words like "get," indicating broad, neutral conversation.
- **Cluster 1** (17 tweets) and **Cluster 2** (18 tweets) are small, positive clusters dominated by "good morning," suggesting morning greetings.

Table 5: Cluster Summary: Size, Top Word, and Sentiment Distribution

Cluster	Size	Top Word (Count)	Positive (4)	Negative (0)
-1 (Noise)	270	(varied, N/A)	N/A	N/A
0	8674	(get, 442)	0.503	0.497
1	17	(good, 17)	1.0	0.0
2	18	(good, 20)	0.889	0.111
3	842	(im, 866)	0.381	0.619
4	19	(im, 19)	0.368	0.632
5	53	(good, 58)	0.679	0.321
6	37	(thanks, 37)	0.919	0.081
7	11	(im, 16)	0.364	0.636
8	8	(day, 8)	0.625	0.375
9	14	(good, 17)	0.786	0.214
10	7	(im, 7)	0.0	1.0
11	5	(im, 5)	0.8	0.2
12	11	(good, 12)	0.545	0.455
13	5	(im, 7)	0.6	0.4
14	5	(im, 6)	0.6	0.4
15	4	(im, 7)	0.5	0.5

- **Cluster 3** (842 tweets) leans negative (61.9%) with "im" and "sorry," reflecting personal or apologetic content.
- Smaller clusters (4–15) show diverse themes (e.g., "thanks" in Cluster 6, "im" in Clusters 7, 13, 14, 15) with varying sentiment, from fully negative (Cluster 10) to mixed (Cluster 15).
- **Noise (-1)** (270 tweets, 2.7%) represents outliers with varied content, lacking a dominant theme or sentiment, possibly due to sparse or unique tweets.

4.5 Comparative Analysis: OPTICS vs. Euclidean DBSCAN

4.5.1 Comparison of Clustering Methods

We compared the performance of OPTICS (Ordering Points To Identify the Clustering Structure) with the initial Euclidean DBSCAN implementation on the Sentiment140 dataset subsample. The evaluation metrics are summarized below:

4.5.2 Interpretation

- **Silhouette Score**: The Euclidean DBSCAN achieves a positive Silhouette Score of 0.609, indicating well-separated clusters, whereas OPTICS

Table 6: Example Tweets for Each Cluster

Cluster	Example Tweet
-1 (Noise)	(No specific example available, represents outliers) ...
0	gunsandbutter12 o yeahI remeber u weren't happy bout that ...
1	AAA_Amerah - good morning, Amerah! ...
2	good night & gute nacht ...
3	OMG im so tired!! that's why I don't go out during the work week. Im lagging! I need wings!...
4	I'm such a wimp ...
5	Nah its all good yea that's fine. Just good for a laugh. ...
6	@CathyCarter thanks cathy ...
7	@skillzva awwwwwwwww..... you are not!! Im on my way!! ...
8	What a gloomy day! ...
9	It's not as good as i hoped ...
10	@itsTONYagain im sorry...
11	@AnTruong1 no ....I'm going to sd tonite!...
12	@TiffanyTrammel oh no that's not good ...
13	im being serious. I have many stories. Cso believes me...
14	I'm out... *kicks Twitter* ...
15	I'm back guys :3 wee. I'm ready to tweet again ...

yields a negative score of -0.533, suggesting overlapping or poorly defined clusters. This may reflect OPTICS's sensitivity to noise and its hierarchical nature, which does not enforce strict cluster boundaries.

- **Calinski-Harabasz Index**: DBSCAN's higher score (1436.657) compared to OPTICS (34.325) indicates better-defined cluster separation and compactness, reinforcing DBSCAN's effectiveness for this dataset.

- **Noise Points**: DBSCAN identifies only 2.7% noise points, suggesting robust clustering, while OPTICS detects 19.2% noise, indicating a tendency to classify more points as outliers, possibly due to its adaptive reachability distance approach.

- **Overall**: Euclidean DBSCAN outperforms OPTICS in this context, likely due to the dataset's structure favoring density-based clustering with a fixed ϵ. OPTICS may be more suitable for datasets with varying densities or complex hierarchies, but its performance here suggests the need for parameter tuning (e.g., ξ or minimum samples).

Table 7: Comparison of Clustering Performance Metrics

Metric	Euclidean DBSCAN	OPTICS
Silhouette Score	0.609	-0.533
Calinski-Harabasz Index	1436.657	34.325
Noise Points (%)	2.7	19.2

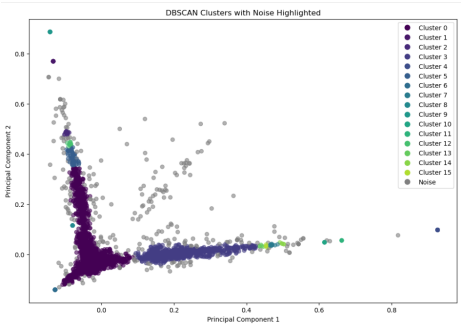

Figure 14: Tweet data after clustering

4.6 Visualization and Analysis

- **Cluster Visualization**: Scatter plots of PCA-reduced data colored by cluster labels.
- **Word Clouds**: Generated for the first three clusters (excluding noise) to highlight prevalent themes.
- **Cluster Characteristics**:
  - Analyzed top 10 words and sentiment distribution for each cluster.

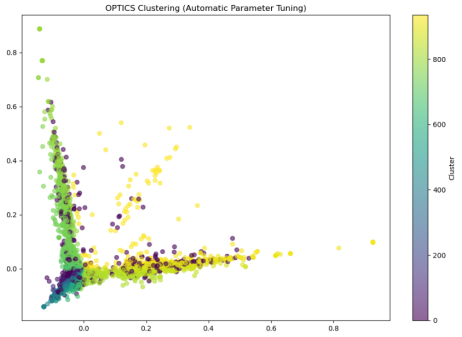

Figure 15: Tweet data after clustering by optics clustering

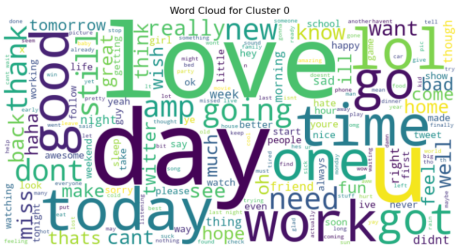

Figure 16: Word cloud for Cluster 0

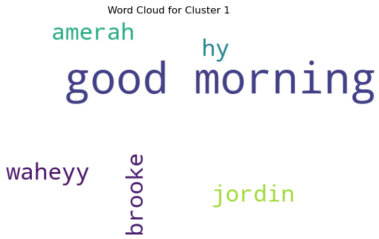

Figure 17: Word cloud for Cluster 1

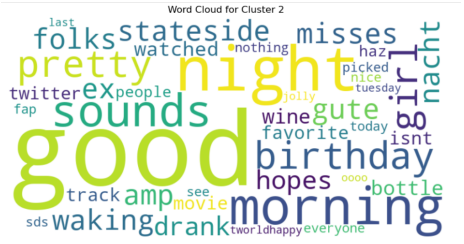

Figure 18: Word cloud for Cluster 2

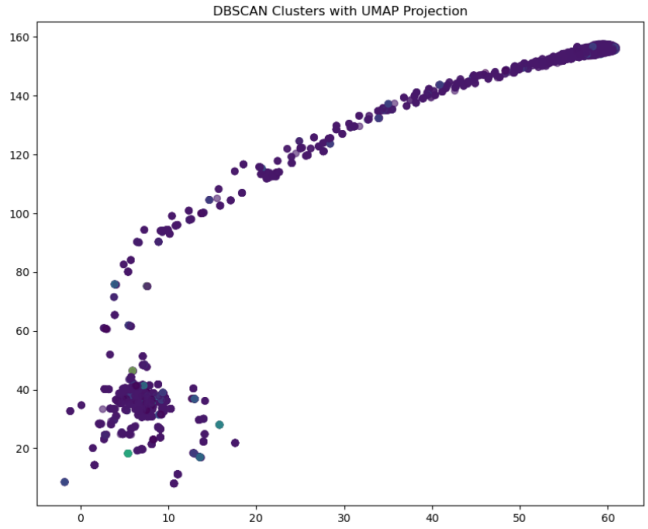

Figure 19: DBSCAN clusters with UMAP projection

  - Example tweets provided for qualitative insight.

- **UMAP Projection**: Optional UMAP reduction for an alternative 2D visualization of clusters.

4.7 Interpretation of Key Findings

- Clusters reflect distinct topics or sentiment patterns (e.g., positive vs. negative tweets), as indicated by word clouds and sentiment distributions.
- High noise percentage suggests diverse or sparse tweet content, requiring further preprocessing or feature engineering.
- The ϵ value of 0.092 effectively balanced cluster formation and noise detection, though adjustment may refine results.

5 Task 4: Association Rule Mining

5.1 Dataset Description

The **Market Basket** dataset contains grocery store transactions, with 464 records of 22 items such as bread, milk, eggs, and others. Each transaction is represented as a binary-encoded vector where 1 indicates the presence of an item and 0 indicates its absence.

5.2 Preprocessing

- Converted csv dataset into a matrix format.
- Visualised data by countplot for every item.

5.3 Implementation Details

We implemented Association Rule Mining from scratch using the Apriori algorithm, a classical method for discovering frequent itemsets and deriving association rules. The process is as follows:

- **Apriori Algorithm Method**:
  1. **Initialization**: Start with 1-itemsets and calculate their support.
  2. **Candidate Generation**: Generate candidate k-itemsets by joining frequent $(k-1)$-itemsets and pruning those with infrequent subsets.
  3. **Support Counting**: Compute the support for each candidate itemset.
  4. **Frequent Itemset Selection**: Retain itemsets exceeding the minimum support threshold.
  5. **Rule Generation**: Derive rules from frequent itemsets by computing confidence and lift, filtering those above the minimum confidence threshold.
- **Metrics**:
  - **Support**: Measures the proportion of transactions containing an itemset.

    $$\text{Support}(A \Rightarrow B) = \frac{\text{Number of transactions containing both } A \text{ and } B}{\text{Total number of transactions}}$$

  - **Confidence**: Measures the reliability of the rule.

    $$\text{Confidence}(A \Rightarrow B) = \frac{\text{Support}(A \cup B)}{\text{Support}(A)}$$

– **Lift**: Measures the strength of a rule over random co-occurrence.

$$\text{Lift}(A \Rightarrow B) = \frac{\text{Confidence}(A \Rightarrow B)}{\text{Support}(B)}$$

- **Thresholds**: Minimum support = 0.05, minimum confidence = 0.5.

5.4 Results

- Maximum support: $\{Banana\}$ (Support: 0.448)
- Number of rules generated: 4536
- Top rule: $\{Butter, Egg\} \Rightarrow \{Bacon, Cheese\}$ (Support: 0.082, Confidence: 0.507, Lift: 2.261)
- **Insights**: Strong associations are observed among staple items such as milk and bread, suggesting frequent co-purchasing behavior.

Table 8: Top 10 Frequent Itemsets

Itemset	Support
$\{Banana\}$	0.448
$\{Cheese\}$	0.444
$\{Bacon\}$	0.431
$\{Hazelnut\}$	0.420
$\{HeavyCream\}$	0.416
$\{Honey\}$	0.416
$\{Carrot\}$	0.414
$\{Bread\}$	0.407
$\{Apple\}$	0.405
$\{ShavingFoam\}$	0.405

Table 9: Top 10 Association Rules by Lift

Rule	Support	Confidence	Lift
$\{Butter, Egg\} \Rightarrow \{Bacon, Cheese\}$	0.082	0.507	2.261
$\{Butter, Banana\} \Rightarrow \{Bacon, Cheese\}$	0.091	0.506	2.258
$\{Toothpaste, Bread, Onion\} \Rightarrow \{Butter\}$	0.060	0.718	1.915
$\{Bacon, Cheese, Onion\} \Rightarrow \{Butter\}$	0.075	0.714	1.905
$\{Banana, Apple, Milk\} \Rightarrow \{Onion\}$	0.060	0.718	1.893
$\{Cheese, Bread, Onion\} \Rightarrow \{Butter\}$	0.062	0.707	1.886
$\{Bacon, Flour, Toothpaste\} \Rightarrow \{Butter\}$	0.052	0.706	1.882
$\{Sugar, Cheese, Onion\} \Rightarrow \{ShavingFoam\}$	0.060	0.757	1.868
$\{Butter, ShavingFoam, Banana\} \Rightarrow \{Bacon\}$	0.080	0.804	1.866
$\{Bacon, ShavingFoam, Hazelnut\} \Rightarrow \{Butter\}$	0.084	0.696	1.857

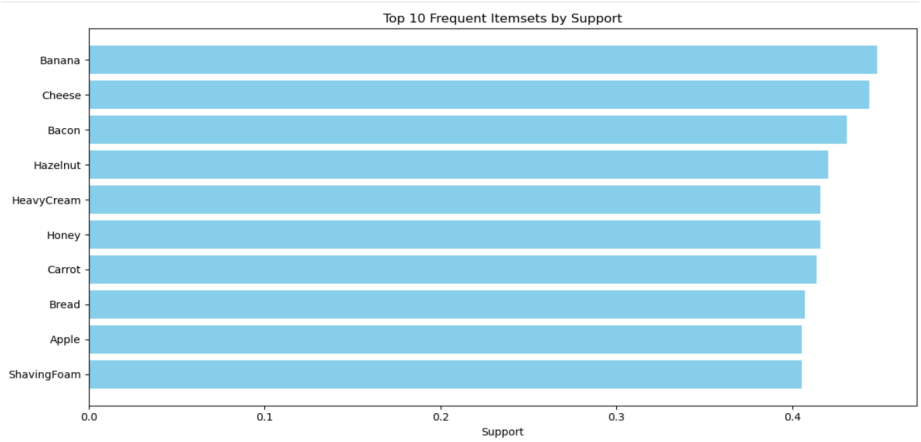

Figure 20: Top 10 frequent item by support

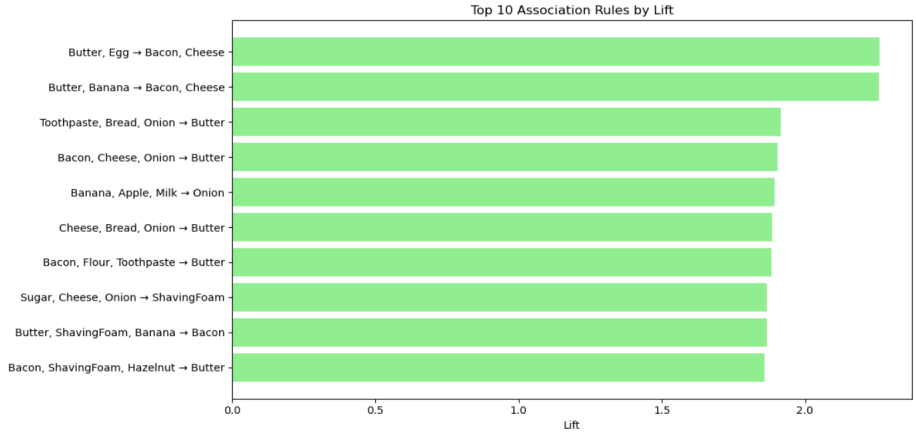

Figure 21: Top 10 Association rules by lift

5.5 Interpretation of Key Findings

- The high lift value (e.g., 2.261 for $\{Butter, Egg\} \Rightarrow \{Bacon, Cheese\}$) indicates a strong positive association, suggesting that customers buying butter and eggs are over twice as likely to buy bacon and cheese compared to random chance. This is valuable for cross-promotional strategies.
- Rules with confidence above 0.5 (e.g., 0.507 for the top rule) are reliable predictors, supporting targeted marketing campaigns for staple goods.
- Frequent itemsets with support around 0.03–0.04 highlight popular combinations, aiding inventory and shelf placement decisions.

6 Conclusion

This lab demonstrated the application of PCA and data mining techniques across diverse datasets. PCA enabled effective dimensionality reduction, enhancing algorithm performance. K-Means segmented customers, Hierarchical clustering structured publications, DBSCAN identified tweet clusters, and Association Rule Mining revealed shopping patterns.