Social Cognition, Vol. 43, No. 3, 2025, pp. 194–216
 https://doi.org/10.1521/soco.2025.43.3.194

From Data to Discovery: Unsupervised Machine Learning's Role in Social Cognition

Jonathan E. Doriscar
Department of Psychology and Institute for Policy Research, Northwestern University

Michalis Mamakos
Department of Management, Kellogg School of Management, Northwestern University

Sylvia P. Perry
Department of Psychology and Institute for Policy Research, Northwestern University

Tessa E. S. Charlesworth
Department of Management, Kellogg School of Management, Northwestern University

The study of how cognition and society interact is a complex endeavor that demands multiple methods and tools. Yet research in social cognition has only begun to capitalize on unsupervised machine learning (UML) tools that can uncover hidden patterns in data. In this tutorial, we introduce UML as a complementary approach to traditional statistical methods. We illustrate four methods (K-means clustering, Density-Based Clustering of Applications With Noise [DBSCAN], Principal Component Analysis [PCA], and Market Basket Analysis) applied to data from Project Implicit and the Implicit Association. We show how UML can identify patterns and relationships that conventional methods might overlook. Throughout, we provide clear (and openly available) code and highlight important researcher decision points in implementing UML in social cognition work. By bringing the advances of UML into social cognition, we will be better equipped to tackle larger, more diverse, or multilevel data sets that reveal the complexities of our social world.

Keywords: unsupervised machine learning, social cognition, Implicit Association Test, clustering algorithms, Principal Component Analysis, Market Basket Analysis

The study of social cognition aims to understand how cognitive processes shape the perception, retention, and utilization of information about social groups (e.g., stereotypes; Fiske, 1993). Social cognition research has a lengthy history

Address correspondence to Jonathan Doriscar, Swift Hall, 2029 Sheridan Rd., Room 412, Evanston, IL 60208. E-mail: JonathanDoriscar2027@u.northwestern.edu

(Higgins & Bargh, 1987), but rapidly evolving computational approaches are now reshaping how we study social cognitive phenomena (e.g., by using natural language processing and agent-based modeling; Charlesworth et al., 2021; Kosinski, 2019). Keeping up with these computational approaches may seem daunting, but they are grounded in familiar machine learning (ML) tools that many researchers already understand. Indeed, ML has been widely used in social cognitive research, including to predict policy support for sexual minorities (Hatzenbuehler et al., 2020), classify hate speech on social media (Davani et al., 2023), and identify patterns of regional intergroup bias (Hehman et al., 2021). However, tutorials on ML for social scientists remain rare and primarily focus on supervised ML (SML, used for prediction; Yarkoni & Westfall, 2017; see also Pargent et al., 2023).

In this tutorial, we therefore review ML principles and introduce common unsupervised ML (UML, primarily used for uncovering patterns in data sets) tools to reduce barriers for social cognition researchers in using UML to study the link between cognitive processes and social behavior. Of note, our goal is not to propose UML as a replacement for hypothesis-driven, theory-based approaches, but rather to show how UML can complement traditional methods by uncovering new patterns, highlighting gaps in existing theories, and inspiring new research questions. This tutorial aims to guide researchers toward a flexible approach that leverages both data-driven discoveries and theory-driven insights—a blend of top-down and bottom-up perspectives.

We organize the remainder of the article as follows: We briefly review SML as a contrast for understanding UML. We then review four UML tools (K-means clustering, Density-Based Clustering of Applications With Noise [DBSCAN], Principal Component Analysis [PCA], and Market Basket Analysis [MBA]). Each tool is applied to large-scale survey data on race attitudes (Project Implicit; Xu et al., 2024), with all R code and data provided via OSF.

SUPERVISED MACHINE LEARNING

SML involves predicting a *target variable*, with a goal of finding the best combination of *features* to predict this target (i.e., the model is "supervised" by the target variable). For example, one might try to predict implicit pro-White/anti-Black bias (i.e., Implicit Association Test [IAT] scores) using input variables like explicit attitudes or political ideology. SML tools encompass *regression*, for continuous outcomes like IAT scores, and *classification*, for categorical outcomes like "high bias" or "low bias." Data are divided into training (typically ~60%–80% of the data) and testing sets (typically ~20%–40% of the data). The model "learns" relationships from the training set, and then evaluates out-of-sample accuracy on the testing set using metrics like root-mean-square error (RMSE), mean-absolute error (MAE), or R^2. In choosing a final SML model, the researcher has to balance between *underfitting*, which oversimplifies the model and leads to systematic *bias*, versus *overfitting*, which overcomplicates the model and reduces generalizability (Yarkoni & Westfall, 2017). (See Table 1 for definitions of terms used in SML.)

TABLE 1. Glossary of Terms Used in Supervised Machine Learning

Term	Definition
Features	Independent variables or predictors in a data set. For example, features could include explicitly measured attitudes, U.S. state, political ideology, and other relevant demographic variables used to predict IAT D-scores.
Target variables	The outcome variables that the model aims to predict. For example, the target variable could be the IAT D-score, which measures implicit pro-White/anti-Black attitudes.
Classification	A type of SML task aimed at predicting categorical outcomes. For example, classification could involve predicting categories like "high bias," "moderate bias," or "low bias" based on the IAT D-scores and other features.
Overfitting	When a model learns the noise in the training data rather than the underlying pattern, leading to poor generalization to new data. For instance, if the model is too tailored to the specific IAT D-scores and explicit attitudes data set, it may perform poorly on new, unseen data.
Underfitting	When a model is too simple to capture the underlying pattern in the data, resulting in poor performance on both the training and test data sets. For example, underfitting occurs in the context of predicting IAT D-scores and explicit attitudes if the model fails to accurately capture the complex relationships between variables, such as the subtle influences of political ideology or demographic factors, leading to inaccurate predictions across different data sets.
Flexibility	The ability of a model to fit a wide variety of data patterns. More flexible models can capture more complex relationships but may be more difficult to interpret. For example, a Random Forest model might capture complex interactions between political ideology and IAT D-scores, but its results may be less interpretable than those from a linear model.
Interpretability	The extent to which a model's predictions can be understood and explained. Simple models like linear regression are usually more interpretable, allowing researchers to understand how explicit attitudes and political ideology predict IAT D-scores.
Bias	The tendency of a model to make systematic errors in one direction. High bias can lead to underfitting, where the model is too simple to capture the underlying pattern in the data, such as not fully accounting for the relationship between political ideology and IAT D-scores.

Note. IAT = Implicit Association Test.

UNSUPERVISED MACHINE LEARNING

Unlike SML, which predicts specific outcomes (e.g., IAT D-scores), UML identifies natural clusters or structures in the data without requiring a predefined target (or dependent) variable. Thus, rather than having a single focal variable "supervising" the process, UML enables researchers to discover hidden relationships across all variables simultaneously.

The field of social cognition grapples with several complex questions that traditional methods alone often struggle to answer—questions that UML methods are particularly well positioned to examine. For example, understanding how individuals' multiple intersecting identities (e.g., race, gender, class) shape their own and others' social cognitive processes is a key challenge (Settles & Buchanan, 2014). To address this, clustering methods like hierarchical clustering or OPTICS

could reveal how individuals naturally group across these dimensions, offering insights that challenge or refine existing theories. Similarly, variability in prejudice-reduction intervention effectiveness remains an open question in the field (Paluck et al., 2021). Topic modeling approaches, such as Latent Dirichlet Allocation (LDA) and Structural Topic Modeling (STM), could analyze participants' open-ended responses to identify nuanced patterns in how interventions are perceived. Even within traditionally studied groups, researchers have noted significant diversity, such as racial heterogeneity (Martinez, 2024). Graph-based techniques like the Louvain method could map these within-group dynamics to uncover novel structures of identity or shared experience. These examples highlight how UML complements theory-driven approaches while enabling researchers to explore new dimensions of social cognition.

This tutorial focuses on introducing both common (e.g., K-means, PCA) and less familiar (e.g., DBSCAN, MBA) UML methods to provide researchers with a foundation for expanding their analytic toolkit. Specifically, we showcase techniques to cluster participants based on key measures, such as attitudes or demographics (K-means, DBSCAN), reduce complex data sets into principal components (PCA), and identify co-occurrence patterns at the item level (MBA). In the following sections, we provide step-by-step guidance on each UML method, emphasizing key decision points—such as determining the optimal number of clusters or interpreting component loadings. We also include R code for implementing each method, shown in figures with executable code in black and comments in green (not treated as R code). For additional support, a set of YouTube videos (linked in the supplemental materials and available on GitHub: https://github.com/jedoriscar/Unsupervised-Machine-Learning.git) walks users through the implementation of all discussed methods.

K-MEANS AND DBSCAN CLUSTERING

Social cognition researchers often aim to identify meaningful patterns within populations, especially across complex attitudes, beliefs, and behaviors. However, traditional analyses typically rely on predefined categories, such as political affiliation or socioeconomic status. In contrast, clustering enables the discovery of nuanced categories that emerge from intersections of relevant variables—such as implicit and explicit attitudes or political ideology. For example, imagine you are a prejudice researcher developing an intervention. You recognize that the effectiveness of your intervention may vary across participants, but you find it challenging to explain this variability using your current variables. You know from past research that people's explicit and implicit attitudes toward Black Americans are associated with political ideology (Charlesworth & Banaji, 2022; Nosek et al., 2007). However, traditional analyses often struggle to uncover the complex patterns that arise when politics, implicit attitudes, and explicit attitudes are considered simultaneously. Clustering is useful here because it can uncover natural patterns among combinations of these attitudes and ideologies, revealing distinct psychological profiles (e.g., individuals with liberal ideologies, pro-White/anti-Black implicit

attitudes, and anti-Black explicit attitudes). Identifying such profiles could help tailor interventions more precisely, enhancing their effectiveness for different subsets of participants (Gaertner & Dovidio, 2000).

Notably, clustering can be used for both empirically driven data mining and as a theoretically guided testing tool, allowing researchers to compare discovered clusters with patterns predicted by existing theories. In other words, clustering, as with all UML tools, is most useful when used with theoretical predictions and researcher expertise to ensure that the results are grounded and generalizable to other settings.

Finding Clusters. Clusters are identified by grouping observations that are most similar to each other. Clustering algorithms do this by using a *similarity measure* (e.g., correlations) or a *distance measure* (e.g., Euclidean distance) between pairs of observations or variables (e.g., respondents, if clustering people; or variables, if clustering variables). Pairwise similarities are then summarized in a matrix. Various clustering algorithms (e.g., K-means/modes, DBSCAN) are used to find optimal groupings; the choice of algorithm depends on data type (e.g., numeric vs. discrete), and the hypothesis (e.g., a hypothesized hierarchical structure).

In this tutorial, we focus on K-means and DBSCAN clustering algorithms because they are widely used yet offer distinct approaches to clustering (Malik & Tuckfield, 2019): K-means is useful for its simplicity and efficiency with large data sets, while DBSCAN is better suited for identifying clusters of arbitrary shapes and effectively handling noise. For a brief summary of their respective strengths, limitations, and usage guidelines, see Table 2.

K-Means Clustering. K-means is a clustering algorithm that groups observations into K clusters by assigning each observation to the nearest centroid (i.e., the average position within a cluster) and iteratively adjusting centroids to minimize distances. Standardizing variables before clustering ensures that no single variable skews the process, making distance calculations consistent. While methods like elbow plots and silhouette scores (Kodinariya & Makwana, 2013) help suggest an optimal K, the choice ultimately rests on researcher expertise. Automated approaches (Roy, 2021) can support this selection, but they lack the theoretical context and interpretational abilities of researchers. Because K-means is sensitive to initial centroid placement, multiple runs (i.e., different initial random positions of the centroids) or robustness checks are recommended. (See Table 3 for definitions of terms used in K-Means Clustering.)

Implementing K-Means Clustering. To implement K-means clustering, we follow four steps: (a) select and standardize the variables for analysis, (b) determine the optimal number of clusters (K), (c) apply the K-means algorithm and perform cross-validation, and (d) interpret the results. For Step 1 in our example, we chose to cluster people based on their responses to three variables: Political Identity (7-point Likert scale with higher scores indicating liberalism), IAT bias (with higher D-scores indicating pro-White/anti-Black implicit bias), and a Feeling Thermometer (11-point "thermometers," with higher scores indicating warmer

TABLE 2. Strengths, Weaknesses, and Usage Guidelines for K-Means and DBSCAN

Method	Strengths	Weaknesses	When to Use	When Not to Use
K-means	Simple to interpret, computationally efficient, and works well with large datasets and spherical clusters.	Sensitive to initial centroid selection, may struggle with nonspherical clusters. Requires predefined choice of K, which can influence clustering results.	Ideal when the researcher has a rough idea of the number of clusters (K) and expects clusters to be spherical and relatively balanced.	Not suitable if clusters are expected to be nonspherical, highly imbalanced, or if K is unknown and difficult to estimate. Also less effective for clusters with varying densities and noncontinuous data.
DBSCAN	Effective for identifying clusters of varying shapes and sizes, can handle noise points, and doesn't require the number of clusters (K) to be specified.	Requires careful parameter tuning, especially for epsilon (the neighborhood radius), and may struggle with high-dimensional data. The choice of epsilon and minimum points (minPts) is data-sensitive and requires researcher judgment.	Useful when there are clusters of varying shapes and noise in the dataset, or when clusters do not form simple geometric shapes. Ideal for datasets where the number of clusters is unknown.	Less effective for high-dimensional data or when clusters have very similar densities. Also challenging when the dataset lacks clear boundaries, as DBSCAN may incorrectly label noise points.

TABLE 3. Glossary of Terms Used in K-Means Clustering

Term	Definition
Centroid	In clustering methods like K-means, the centroid is the average position of all data points in a cluster in multidimensional space. For example, the centroid could represent the typical profile of individuals within a cluster based on their IAT scores and explicit attitudes, such as those with moderate explicit bias but low implicit bias.
Within-cluster sum of squares	Within-Cluster Sum of Squares measures the compactness of clusters by summing the squared differences between each participant's responses and the cluster centroid. Lower Within-Cluster Sum of Squares could indicate more cohesive clusters of individuals with similar IAT scores and attitudes.
Cross-validation	Cross-validation is the process of refining a model or clustering solution through successive iterations until it stabilizes. For example, this would involve running the clustering algorithm multiple times to ensure that the identified clusters consistently capture meaningful groupings of implicit and explicit biases.
Distance measure	A distance measure, such as Euclidean distance, quantifies how similar or different data points are in a multidimensional space. For example, it could help determine the similarity between participants based on their responses to implicit and explicit attitude measures, aiding in identifying groups with similar biases.

Note. IAT = Implicit Association Test.

feelings toward Black people). In addition, we standardize the variables so that each variable contributes the same amount to the clustering process.

Step 2 is to decide on the number of clusters (K). While the choice of parameter can be guided by previous literature, it can be challenging to determine for novel data sets with many variables. The *elbow method* is a visualization tool that helps identify the optimal number of clusters by minimizing within-cluster distance (such that the data points within a given cluster are more similar to each other). This method (illustrated in Figure 1) involves plotting the total *within-cluster sum of squares* (WSS; i.e., the average distance of the data points from their centroids) against the number of clusters. The optimal K is indicated by the elbow point, where the WSS shifts from rapidly decreasing to a slower decline, indicating that additional clusters provide limited explanatory value).

In the R code (Figure 1a), we calculate the total WSS for different K-values and then visualize the elbow plot. Figure 1b shows the tradeoff between clustering quality (WSS) and parsimony (number of clusters). The elbow method addresses this trade-off by identifying the point where incremental improvement in WSS diminishes as the number of clusters increases. In our example, because WSS decreases more slowly after K = 5 (Figure 1b), we select this number of clusters. Researchers should look for a noticeable "bend" in the plot, although this may not always be sharply defined, as in our example. To aid interpretation, additional criteria such as the ratio of between-cluster variance to within-cluster variance can provide guidance. Here, we chose K = 5 (i.e., 5 latent clusters of respondents), which gave the highest ratio of between-cluster to within-cluster variance. Specifically, the ratio was 40.9% for K = 3 and 60.9% for K = 5. This K-selection process helps balance under- and overfitting by minimizing within-cluster variance while also avoiding the overfitting that can occur with too many clusters.

In Step 3, we apply the K-means algorithm to calculate the distance between individual data points (i.e., respondents) and cluster centroids using Euclidean distance. We then cross-validate with 25 random cluster coordinate assignments to ensure that they remain consistent regardless of the initial cluster assignments (see Figure 2).

Interpreting K-Means Clustering. Step 4 in the clustering process is interpretation. In Table 4, we highlight the five cluster means identified through the K-means algorithm, showing how respondents in each cluster scored, on average, across our three key variables. In Figure 3, we visualize the shape of these clusters on a two-dimensional plot. Although the clustering occurs across multiple dimensions, the two-dimensional plot provides a simplified representation to aid interpretation and communication. Cluster 1 exhibits a relatively low pro-White/anti-Black bias, slightly liberal political identity, and relatively cold explicit attitudes toward Black people. Cluster 2 is conservative, shows a slight pro-White/anti-Black bias, and has cold feelings toward Black people. Cluster 3 is liberal, has a pro-White/anti-Black bias, and has warm explicit feelings toward Black people. We find that Cluster 4 has a pro-Black/anti-White bias, is slightly liberal, and has warm feelings

A

```
# Elbow Method: Determines the optimal number of clusters by plotting #WSS vs. number of clusters (K)
# The "elbow" point where WSS starts to decrease slowly indicates the #optimal K.
# Create a function to calculate total WSS for different K values
# This function iterates over a range of cluster numbers (1 to max_k) to calculate WSS for each K.
# WSS (Within-Cluster Sum of Squares) quantifies the compactness of the clusters; lower WSS values indicate tighter clusters.
# - Uses `nstart = 25` to initialize k-means clustering with 25 random starting points to improve the stability of results.
wss <- function(data, max_k) {
  sapply(1:max_k, function(k) {
    kmeans(data, centers = k, nstart = 25)$tot.withinss
# Extract total WSS for each K
  })
}
# Calculate WSS for K values from 1 to 10
wss_values <- wss(iat_numeric_scaled, max_k = 10)
# Plot the Elbow Method
fviz_nbclust(iat_numeric_scaled, kmeans, method = "wss") + # Generate WSS plot using `factoextra`
  geom_vline(xintercept = which.min(diff(wss_values)), linetype = 2) + # Add a dashed vertical line at the elbow point
  labs(title = "Elbow Method for Optimal K") # Add a title to the plot
```

B

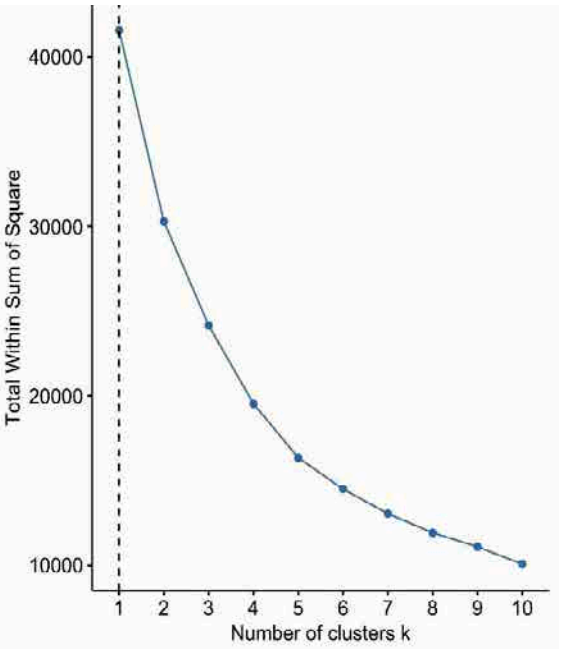

FIGURE 1. Using the Elbow Method to determine the optimal number of clusters. This figure demonstrates the implementation and output of the Elbow Method to identify the optimal number of clusters (K) in K-means clustering. Panel A shows the R code used to calculate the total within-cluster sum of squares (WSS) for a range of cluster numbers (1 to 10 in this example). Panel B presents the WSS values plotted against the number of clusters (K). The "elbow point," where the rate of decrease in WSS slows, indicates the optimal number of clusters. This approach allows researchers to balance cluster compactness with the number of clusters. Panel A: Code Implementation of the Elbow Method; Panel B: Plot of Within-Cluster Sum of Squares (WSS) Against Number of Clusters (K).

toward Black people. Cluster 5 has a pro-White/anti-Black bias, is conservative, and has the most warm feelings toward Black people.

Social cognition theories can be used to guide interpretation of these clusters of respondents. Clusters 3 and 5 align with the concept of aversive racism, where individuals hold underlying prejudices (revealed by their IAT bias) that conflict with their egalitarian self-concept (Dovidio & Gaertner, 2004; Pearson et al., 2009). However, their distinct political identities (Cluster 3 is liberal, Cluster 5 is conservative) suggest that the origins of their aversive racism may differ, necessitating tailored approaches for anti-bias education (Vitriol & Banaji, 2024; Vitriol & Moskowitz, 2021). Cluster 4 has a pro-Black/anti-White bias and warm feelings toward Black people. Researchers interested in understanding the factors associated with more positive racial attitudes could explore what differentiates this group from others. Identifying such factors may provide insight into the conditions under which individuals develop more egalitarian attitudes. Clusters 1 and 2

```
# The algorithm uses Euclidean distance by default to measure the distance between data points and their cluster centroids.
set.seed(1234)
# Setting a random seed ensures that the cluster assignments are consistent each time the code is run,
# as k-means clustering involves a random initialization of centroids.
km.res <- kmeans(iat_numeric_scaled, 5, nstart = 25)
# Perform k-means clustering on the scaled data ('iat_numeric_scaled').
# - `centers = 5`: Specifies the number of clusters (K) to partition the data into.
# - `nstart = 25`: Runs the algorithm 25 times with different random centroid initializations
#   and selects the clustering solution with the lowest within-cluster sum of squares.
print(km.res)
# Output of `kmeans()`:
# - `centers`: The centroids (mean coordinates) of the clusters in the data space.
# - `totss`: Total sum of squares (TSS), measuring the total variance in the dataset.
# - `withinss`: A vector of within-cluster sum of squares for each cluster, measuring compactness.
# - `tot.withinss`: Total within-cluster sum of squares across all clusters, used for evaluating clustering fit.
# - `betweenss`: Between-cluster sum of squares, capturing the separation between clusters.
# - `size`: Number of data points assigned to each cluster, indicating cluster sizes.
```

FIGURE 2. R Code for performing K-means clustering. This figure illustrates the R code used to perform K-means clustering on the scaled data set. The code specifies the number of clusters (K = 5) and runs the algorithm 25 times with different random initializations to ensure a stable solution with the lowest total within-cluster sum of squares. The output of the kmeans() function includes cluster assignments, centroids, and measures of fit, such as total within-cluster sum of squares (compactness) and between-cluster sum of squares (separation).

seem to represent explicitly racist individuals, but they potentially differ in their motivations, as suggested by their differing political identities.

Density-Based Clustering of Applications With Noise (DBSCAN). In our K-means example, Cluster 3 displayed overlap with other groups, likely because it included multiple subgroups (see Figure 3 for this overlap). This scenario suggests that we might find a better solution using the DBSCAN algorithm. Unlike K-means, which requires specifying the number of clusters (K) and assumes spherical cluster shapes, DBSCAN identifies clusters based on the density of points and does not require a predefined number of clusters. This makes DBSCAN particularly useful when clusters are nested within one another and have arbitrary shapes, or when data sets

TABLE 4. K-Means Clustering: Means of the Key Variables Across Respondents Assigned to Each of the 5 Clusters

Variable	Cluster 1 (*n* = 2303)	Cluster 2 (*n* = 2771)	Cluster 3 (*n* = 3070)	Cluster 4 (*n* = 2636)	Cluster 5 (*n* = 3075)
IAT D-score	0.171	0.305	0.412	−1.381	0.368
Political Identity	0.811	−0.906	0.854	0.351	−0.944
Feeling Thermometer (Black)	−1.07	−1.14	0.628	0.575	0.712

Note. IAT = Implicit Association Test. All variables are *Z*-scored. Positive Political Identity scores indicate liberalism, positive IAT scores indicate pro-White/anti-Black bias, and positive Feeling Thermometer scores indicate warmer feelings toward Black people.

TABLE 5. Glossary of Terms Used in DBSCAN

Term	Definition
Density	In DBSCAN, density refers to the number of data points packed closely together in a given area. For example, density could help identify clusters where participants have similar IAT scores, indicating where attitudes naturally group.
Epsilon (ε)	Epsilon in DBSCAN is the maximum distance between two points to be considered neighbors. For example, with IAT scores, a smaller epsilon groups individuals with very similar scores, while a larger epsilon includes those with broader similarities.
Core points	In DBSCAN, core points are data points that have a sufficient number of neighboring points within the epsilon distance. For example, these points could help define clusters, representing central members of a group with similar implicit and explicit biases in IAT data.

Note. DBSCAN = Density-Based Clustering of Applications With Noise; IAT = Implicit Association Test.

A

```
# Visualize the K-means clustering results
----
# The 'factoextra' package provides an easy
way to create cluster visualizations.
fviz_cluster(km.res,
# - `km.res`: This object contains the
k-means clustering results, including cluster
assignments for each data point.
        data = iat_numeric_scaled,
# The scaled dataset used for clustering
        geom = "point",
# Each data point is represented as a dot on
the plot
        ellipse.type = "convex",
# Draw convex hulls around the clusters to
show their boundaries
        ggtheme = theme_minimal(),
# Apply a minimalistic theme for improved
readability
        palette = "jco")
# Use the 'jco' palette for distinct and
visually appealing cluster colors
# - `geom = "point"`: Visualizes individual
data points, helping us identify cluster
membership.
# - `ellipse.type = "convex"`: Convex hulls
encapsulate the points in each cluster,
showing their spatial grouping.
```

B

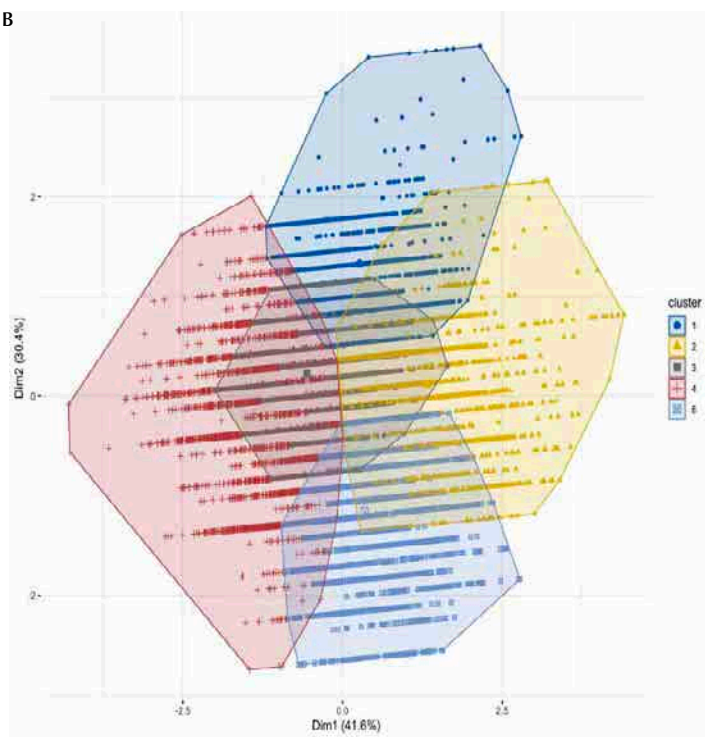

FIGURE 3. Visualizing K-means clustering results. This plot visualizes the K-means clustering results, grouping data points (represented as dots) into clusters based on their assignments. Each convex hull outlines the boundaries of a cluster, while colors and shapes distinguish between different clusters. The plot is displayed in two dimensions, using the first two principal components (Dim1 and Dim2) to project the high-dimensional data. Panel A: Code Implementation for K-Means Visualization; Panel B: Visual Output of K-Means Clustering Results.

have a lot of noise (see Table 5 for a glossary of terms used in DBSCAN). DBSCAN is widely utilized in social science research, such as analyzing clusters of opinions on natural disasters (Mustakim et al., 2021) or grouping consumers based on their geo-tagged social network data (Fan et al., 2021).

The DBSCAN algorithm requires a parameter (typically denoted as ε) that defines the radius within which points are considered neighbors. Points with a sufficient number of neighbors (at least *MinPts*) within this radius become *core points*. The algorithm starts with an arbitrary core point and assigns it as the first cluster, then adds nearby points within the ε radius to the cluster. If a point is close enough to multiple core points, it is added to the cluster. Points not close enough to any core points are considered "noise" and are left unclustered. A second cluster is created when a new core point, with enough neighboring points within ε, is identified outside the reach of the first. This process repeats, with each new core point potentially forming a new cluster if it is sufficiently distant from existing clusters, allowing DBSCAN to identify multiple clusters of arbitrary shapes within the data.

Implementing DBSCAN. To implement DBSCAN, we follow three steps using the same numeric and standardized variables from our K-means example: (a) determine the optimal parameters (e.g., ε, MinPts); (b) apply the DBSCAN algorithm to identify core points (see Figure 4); and (c) interpret the results.

For Step 1 (see Figure 4a for code), we can use plotting (as we did for K-means) to determine the optimal ε value. Specifically, we use k-nearest neighbors (k-NN) plotting (Figure 4b) to identify an elbow point, after which increasing the ε does not significantly improve the model's explanatory value. In our plot, the elbow point for ε is between 0.5 and 0.6. In R (Figure 4a) we iterate with different ε values within that range and find that ε = 0.61 and MinPts = 5 lead to the largest noise reduction as well as similar sample sizes within each of the clusters.

Interpreting DBSCAN. In Table 6, we present the means of key variables across the seven clusters identified using DBSCAN. Clusters 1 and 3 are comparable to Cluster 4 from the K-means example. These clusters are liberal and are characterized by relatively pro-Black explicit attitudes and pro-Black implicit biases (indicated by the negative IAT D-scores in Table 6). Cluster 2, by contrast, can be described as "conflicted racists" because they are liberal, hold anti-Black explicit attitudes, yet still exhibit pro-Black/anti-White implicit attitudes. Clusters 5, 6, and 7 represent overt racists, characterized by highly conservative views, pro-White/anti-Black implicit attitudes, and explicitly anti-Black explicit attitudes.

Comparing these findings with our K-means analysis, we observe similarities in identifying aversive racism and differences in attitudes across political ideologies. However, DBSCAN's ability to handle varying densities provided a more nuanced understanding, especially in identifying two additional clusters beyond those identified through the K-means algorithm. Figure 5 demonstrates DBSCAN's ability to identify clusters with diverse shapes and densities, making it effective for nonspherical data structures that K-means may miss. However, this adaptability

A

```
# Visualizing k-Nearest Neighbors (kNN) Distances ----
# The kNN distance plot is a diagnostic tool used in DBSCAN to determine the optimal value for 'eps' (the neighborhood radius).
# The 'k' parameter corresponds to the number of neighbors (MinPts), which influences how clusters are defined.

# Generate the kNN distance plot for the scaled data
kNNdistplot(iat_numeric_scaled, k = 5) # Set k to 5, aligning with the MinPts parameter in DBSCAN

# Add a horizontal line to highlight the 'elbow point' in the kNN plot
abline(h = 0.55, col = "red", lty = 2) # Set at the chosen 'eps' value (based on the plot inspection)
# The horizontal line indicates the threshold distance for clustering (eps).
# Points below this line are considered part of a dense cluster, while points above may be treated as noise.
```

B

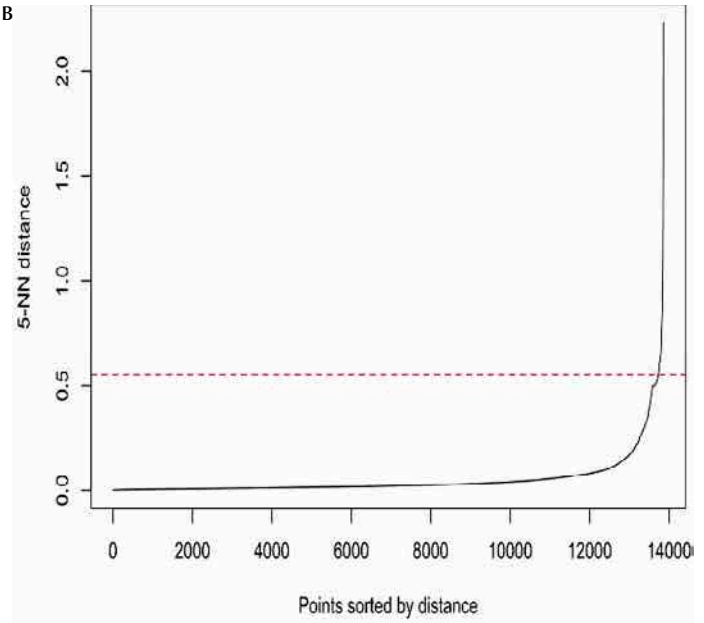

FIGURE 4. Visualizing k-nearest neighbors (kNN) distances for DBSCAN. Panel A shows the R code used to generate the kNN distance plot, a diagnostic tool for selecting the optimal epsilon (eps) value in DBSCAN clustering. Panel B displays the resulting kNN distance plot, with the red dashed line marking the chosen epsilon (eps = 0.55). Points below this threshold are part of dense clusters, while those above may be classified as noise. The "elbow" in the curve highlights the natural inflection point, guiding the selection of an appropriate eps value for effective clustering. Panel A: Code Implementation for K-Means Visualization; Panel B: kNN Distance Plot for DBSCAN Clustering.

can introduce overlap if parameters like ε (the radius) and MinPts (minimum number of points) are not carefully optimized (see Figure 5b for visualization of overlap). Fine-tuning these parameters can reduce overlap, enabling clearer separation between clusters.

DIMENSION REDUCTION AND ASSOCIATION RULES MINING

Data sets in the social sciences are becoming increasingly complex (Kosinski, 2019), with increasing numbers of variables often across massive samples of participants. For example, consider the Project Implicit data sets, with tens of millions of respondents and more than 200 variables that capture overlapping constructs (e.g., multiple indicators of explicit racial attitudes). These data sets are ideal for dimension reduction, which seeks to reduce many variables to a smaller subset while still retaining most of the variability, or for association rule mining, which identifies patterns and relationships among variables without reducing their number.

In this section, we focus on two techniques: (a) Principal Component Analysis (PCA), a widely used dimension reduction method, and (b) Market Basket

TABLE 6. DBSCAN Clustering: Means of the Key Variables Across Respondents Assigned to Each of the 7 Clusters

Variable	Cluster 1 (*n* = 2033)	Cluster 2 (*n* = 1514)	Cluster 3 (*n* = 3557)	Cluster 4 (*n* = 3967)	Cluster 5 (*n* = 460)	Cluster 6 (*n* = 1128)	Cluster 7 (*n* = 1156)
IAT D-Score	−0.188	−0.017	−0.078	−0.001	0.367	.193	0.296
Political Identity	1.39	0.173	0.785	−0.438	−2.27	−1.05	−1.66
Feeling Thermometer (Black)	0.135	−0.039	0.082	−0.008	−0.288	−0.112	−0.083

Note. DBSCAN = Density-Based Clustering of Applications With Noise; IAT = Implicit Association Test. All variables are *Z*-scored. Positive Political Identity scores indicate liberalism, positive IAT scores indicate pro-White/anti-Black bias, and positive Feeling Thermometer scores indicate warmer feelings toward Black people.

A

```
# Density-Based Clustering with DBSCAN ----
# Parameters:
# 'eps' specifies the neighborhood radius for clustering.
# 'minPts' is the minimum number of points required
to form a dense region (cluster).
set.seed(1234) # Set seed for reproducibility in
cluster assignment
db.res <- dbscan(iat_numeric_scaled, eps =
0.61, minPts = 5) # Perform DBSCAN with
specified parameters
# Plot the clustering results without noise points
# Use 'factoextra' to visualize the clustering results,
highlighting actual clusters without noise.
fviz_cluster(list(
  data = iat_numeric_scaled_df_filtered[,
-ncol(iat_numeric_scaled_df_filtered)], #
Exclude the cluster column for plotting
  cluster =
iat_numeric_scaled_df_filtered$cluster), #
Use the cluster assignments
  geom = "point", # Each data observation is shown
as a point
  ellipse.type = "convex", # Draw convex hulls
around each cluster
  ggtheme = theme_minimal(), # Apply a clean,
minimal theme
  palette = "jco" # Use the 'jco' color palette for
clusters)
```

B

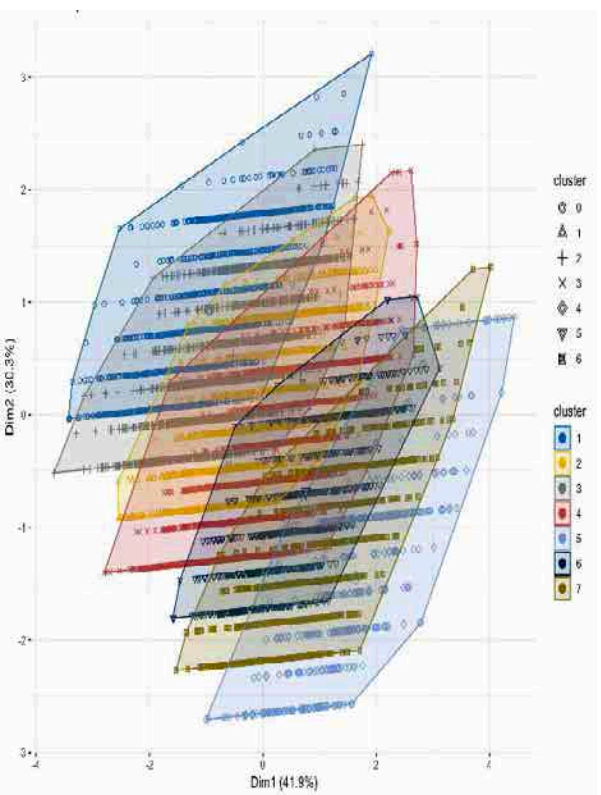

FIGURE 5. DBSCAN clustering: Code implementation and visualization results. This figure demonstrates the R code (Panel A) used to perform density-based clustering with the DBSCAN algorithm and its results visualized in two dimensions (Panel B). In Panel A, key DBSCAN parameters include eps (the neighborhood radius for clustering) and minPts (the minimum number of points required to form a dense region). The output assigns each data point to a cluster, with points labeled as "0" identified as noise. Panel B shows the resulting DBSCAN clusters, with convex hulls outlining the boundaries of each cluster. Noise points (cluster 0) are excluded for clarity. The plot projects high-dimensional data into two dimensions using the first two principal components (Dim1 and Dim2). Panel A: R Code for DBSCAN; Panel B: Cluster Plot for DBSCAN Results.

Analysis (MBA), a technique for association rule mining. PCA and similar methods, like factor analysis, have a long history in psychology (e.g., validating psychological scales; Gruijters, 2019). In contrast, MBA is less commonly applied in psychology but is well-known in fields like economics, where it has been used to analyze geo-tagged data on tourists' spending and vacation experiences (Vavpotič et al., 2021). While PCA helps reduce complexity by summarizing data into principal components, MBA provides an entirely different capability: It identifies co-occurrence patterns and associations between variables, offering unique insights at the item level. Together, these techniques demonstrate how UML can reduce complexity and uncover meaningful patterns in large data sets.

To illustrate these tools, imagine again that you are a social cognition researcher aiming to understand how various factors (e.g., motivation to control prejudice, implicit bias) influence prejudice reduction. You are working with a large data set of millions of participants and hundreds of variables; analyzing all variables would therefore require significant computational power and be unwieldy for interpretation. To address this, you can use PCA to transform numerous correlated variables into a smaller set of uncorrelated components that capture the maximum variance. Or you can use MBA to create association rules that identify combinations of factors (e.g., high motivation to control prejudice and positive feelings toward Black people) that commonly co-occur. You will then be better equipped to identify key factors driving the strongest attitudes (or the weakest attitudes) and develop more targeted interventions.

Principal Component Analysis. In PCA, we begin by calculating either a covariance matrix or a correlation matrix. A correlation matrix is helpful when the variables are on different scales because it standardizes the variables and makes them comparable. Both covariance and correlation matrices reveal how each pair of variables in our data set varies together; higher values indicate greater dependency between variables. PCA is a data-driven technique that does not assume any underlying structure, unlike factor analysis. PCA focuses on capturing the maximum variance in the data through a new set of uncorrelated variables, known as principal components.

Implementing Principal Component Analysis. Implementing PCA follows five steps: (a) select (and standardize) the relevant variables; (b) compute the covariance matrix; (c) perform decomposition on the matrix to obtain *eigenvalues* and eigenvectors, where the eigenvectors determine the principal components and the eigenvalues indicate the variance captured by each component; (d) select the number of principal components to retain; and (e) interpret the results by examining the *loadings* of the original variables on the principal components.

While Steps 1–3 of PCA implementation are data-driven (see Figure 6 for R code), researchers must decide the number of principal components (Step 4) that explain the optimal amount of variance and are most interpretable. As before, visual inspection helps guide this decision. Here, the *scree plot* (see Figure 7b) visually represents the variance explained by each principal component in the PCA.

```
# Perform Principal Component Analysis (PCA) ----
# PCA is a dimensionality reduction technique that transforms the data into a set of
orthogonal components.
# We use the prcomp function to perform PCA on the scaled dataset. Arguments:
#   - center = TRUE: Ensures that the data is centered by subtracting the mean of each
variable.
#   - scale. = TRUE: Scales the data to have a standard deviation of 1, making all variables
comparable.
pca_result <- prcomp(iat_pca_data_scaled, center = TRUE, scale. =
TRUE)
# Summary of PCA results ----
summary(pca_result)
# The summary provides:
#   - Proportion of Variance: Shows the variance explained by each principal component as a
percentage of the total variance.
#   - Cumulative Proportion: Cumulatively adds up the variance explained by the
components, helping to decide how many components to retain.
```

FIGURE 6. Principal component analysis (PCA) code. This figure shows R code for performing PCA using the prcomp function. PCA reduces dimensionality by transforming variables into principal components that capture the most variance in the data. The arguments center the data (subtracting the mean) and scale it (standardizing variables).

A

```
# Print the loadings (principal component vectors) ----
# Loadings indicate how much each original variable
contributes to the principal components.
# These values help interpret the components by showing
the correlation of each variable with the component.
print(pca_result$rotation)
# Store the loadings for further analysis or visualization.
rotations <- pca_result$rotation
# Visualize the variance explained by each principal
component ----
# A scree plot displays the percentage of variance
explained by each principal component.
# The plot helps determine how many components to retain
based on the proportion of variance.
fviz_eig(pca_result, addlabels = TRUE, ylim =
c(0, 50)) +
  labs(
    title = "Scree Plot", # Title of the plot
    x = "Principal Component", # X-axis label
    y = "Percentage of Variance Explained" #
Y-axis label
  )
# Arguments:
#   - addlabels = TRUE: Adds numerical labels to each bar
for clarity.
#   - ylim = c(0, 50): Sets the y-axis limits to ensure proper
scaling.
```

B

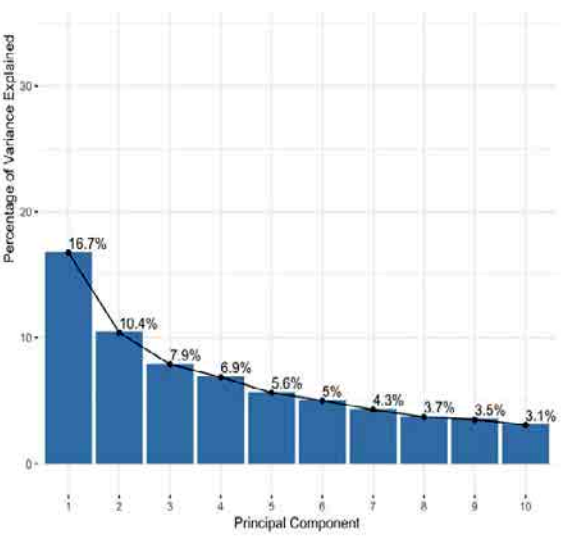

FIGURE 7. Principal component analysis: R code and scree plot. This figure presents R code (Panel A) to extract and visualize key PCA outputs alongside the resulting scree plot (Panel B). The scree plot (Panel B) illustrates the percentage of variance explained by each principal component, guiding the selection of components for further analysis. Each bar represents a principal component, with numerical labels indicating the variance explained. The first few components account for the majority of the variance, with diminishing returns as the components increase. Panel A: R code for PCA Loadings and Variance Visualization; Panel B: Scree Plot of PCA Variance Explained.

In our example (visualized in Figure 7b), Principal Component 1 (PC1) explains 16.7% of the total variance, the highest among all components, followed by PC2 at 10.4%, and PC3 at 7.9%. Each subsequent component explains progressively less variance, with PC10 contributing just 3.1%. The plot shows an "elbow" shape, where the explained variance drops sharply after the first few components and then levels off, indicating that the first few principal components capture most of the variance and additional components would not add to the model fit. This suggests that retaining around the first four principal components is reasonable for dimensionality reduction because they capture a substantial portion of the variance (41.88%) and are still parsimonious.

Interpreting Principal Component Analysis. The loadings for each principal component indicate how much each original variable contributes to explaining the new dimension-reduced components. In this section, we focus on interpreting Principal Component 1 (PC1) for brevity because it explains the largest percentage of variance in the data set.

PC1 is primarily shaped by items from the Motivation to Control Prejudice (mcpr) scale, and generally has high loading from low motivation to control prejudice (and low loading from high motivation to control prejudice). That is, the strongest positive contributors are mcpr Item 5 (loading = 0.28) and mcpr Item 9 (loading = 0.26), both of which reflect low motivation to control prejudice (and specifically a disregard for how others may react to your prejudiced Ibehavior). For example, item 5 states that "Going through life worrying about whether you might offend someone is just more trouble than it's worth," and Item 9 states that "I think that it is important to speak one's mind rather than to worry about offending someone." Conversely, the strongest negative contributors—mcpr Item 13 (loading = −0.29) and mcpr Item 3 (loading = −0.29)—represent individuals who avoid offensive behavior. For instance, Item 13 states, "It bothers me a great deal when I think I've offended someone, so I'm always careful to consider other people's feelings." For a prejudice-reduction scholar, PC1 offers a meaningful new variable to examine how reduced motivation to control prejudice might moderate the effectiveness of an intervention. By interpreting and incorporating this principal component, researchers can simplify their data while generating insights relevant to their theoretical goals.

Market Basket Analysis. In MBA, *rules* are patterns discovered in the data that describe how items (i.e., a specific level or value of a variable such as a 2 on a 7-point scale) co-occur with one another. These rules specify how the presence of certain items (antecedents) imply the presence of other items (consequents). In MBA, you first set a threshold, known as *support*, which specifies the minimum percentage of co-occurrence for an item set (i.e., a collection of items or variables that appear together) to be considered significant. For example, if a low level of religious identity (i.e., 1 on a 7-point scale) co-occurs with a neutral attitude (i.e., 5 on a scale of 0–10) 38% of the time, it would be considered a significant rule if support was set to 30% (because 38% passes the 30% threshold), but not if support were set to 40% (because 38% does not pass that threshold). As before, the decision

TABLE 7. Strengths, Weaknesses, and Usage Guidelines for Principal Component Analysis (PCA) and Market Basket Analysis (MBA)

Method	Strengths	Weaknesses	When to Use	When Not to Use
PCA	Reduces dimensionality, helping to simplify complex datasets, while retaining as much variance as possible. Useful for visualizing high-dimensional data and finding latent patterns.	Limited to continuous, numeric data and can be sensitive to scaling and outliers. Results can be challenging to interpret as principal components are linear combinations rather than original variables.	Suitable for exploratory analysis, dimensionality reduction, and cases where high-dimensional data needs to be visualized or simplified.	Not suitable when the relationships between variables are nonlinear or when the dataset contains categorical variables, as PCA assumes linearity and operates only on numeric data.
MBA	Identifies co-occurrence patterns among variables, ideal for discovering associations and relationships at the item level. Does not require predefined relationships, and the resulting rules are easy to interpret.	Depends heavily on support, confidence, and lift thresholds, which may exclude rare but meaningful associations. Can produce an overwhelming number of rules without careful filtering.	Effective for exploratory purposes when seeking frequent item associations or to understand co-occurring behaviors. Useful for generating rules to inform decision making or to identify prevalent patterns in transactional data.	Less useful for datasets with high dimensionality where co-occurrence patterns may be sparse. Avoid if the focus is on continuous data relationships, as MBA is better suited for categorical or binary data patterns.

of which support to use relies on researcher domain knowledge and testing on the data set. A high support threshold (e.g., only variables that occur together 80% of the time) might exclude meaningful patterns, while a low support threshold (e.g., all variables that occur together 10% of the time) might include too many associations that are not actually meaningful.

Two other outputs of MBA are the *confidence* and *lift* of the generated rules. Confidence measures the likelihood that the presence of one item will result in the presence of another item. For example, if there is 80% *confidence* that a value of 1 on "motivation to control prejudice" co-occurs with a value of 5 on "Black feeling thermometer," it means that 80% of the responses that include Item 1 also include Item 5. *Lift* compares the observed co-occurrence of items to their expected co-occurrence if they were independent (e.g., conceptually similar to a chi-square test of independence). A lift item greater than 1 indicates a meaningful association between items, meaning that the items co-occur more frequently than expected by chance.

Implementing Market Basket Analysis. MBA is implemented in four steps using the same numeric variables from our PCA example along with several additional categorical variables. Although Table 7 highlights that MBA is less suitable for numeric variables, we include them in this example to demonstrate that the method can be applied to numeric variables, albeit less effectively than with categorical data. The steps include: (a) convert the data set into a transactional format suitable for

TABLE 8. Glossary of Terms Used in Principal Component Analysis

Term	Definition
Eigenvalues	In PCA, eigenvalues represent the amount of variance captured by each principal component. For example, larger eigenvalues in IAT data analysis could suggest that a particular component explains a significant portion of variance in implicit and explicit attitudes.
Loadings	Loadings in PCA indicate the contribution of each variable to a principal component. High loadings suggest that a variable is strongly associated with the component.

Note. IAT = Implicit Association Test.

association rule mining; (b) apply the algorithm to generate frequent item sets by setting minimum support and confidence thresholds; (c) extract association rules from these frequent item sets to identify patterns and relationships; and (d) interpret the rules to understand co-occurrences and dependencies within the data structure. See also Tables 8 and 9 for glossary definitions of key terms related to PCA and MBA.

To prepare for MBA, we need to ensure that all relevant variables are in a "transaction" format (see Figure 8a). This format allows us to identify patterns and associations between different responses, similar to how a retailer might identify which

TABLE 9. Glossary of Terms Used in Market Basket Analysis

Term	Definition
Support	In MBA, support quantifies how often a combination of items occurs in the data set. For example, in IAT data, it could measure the prevalence of patterns, such as the proportion of participants with both high explicit and implicit biases.
Rules	In MBA, rules identify combinations of items that frequently co-occur in the data set. For example, in IAT data, these rules could help uncover patterns, such as the tendency for individuals with certain explicit biases to also show implicit bias.
Lift	Lift measures the strength of an association rule in MBA by comparing the observed co-occurrence of items to what would be expected by chance. For example, a lift greater than 1 in IAT data analysis would indicate a strong association between implicit and explicit attitudes, highlighting meaningful patterns.
Confidence	Confidence in MBA measures the likelihood that one item will be present when another is present. For example, in IAT data, high confidence might indicate that participants with a high explicit bias are likely to also show high implicit bias.
Antecedents and Consequents	In MBA, the antecedent represents the initial item(s) that may lead to the occurrence of another item, while the consequent is the item(s) that is likely to occur when the antecedent is present. This relationship is often displayed as "Antecedent => Consequent," with the antecedent on the left-hand side (LHS) and the consequent on the right-hand side (RHS). For example, in IAT data, an antecedent might be a particular explicit attitude while the consequent could be an implicit attitude.

Note. IAT = Implicit Association Test.

A

```
# Perform Market Basket Analysis ----
# Use the Apriori algorithm to identify association rules based on
thresholds for support and confidence.
rules <- apriori(transactions, parameter = list(supp = 0.10,
conf = 0.8))
# 'support' (supp): Minimum fraction of transactions containing the rule
(e.g., 10% of transactions).
# 'confidence' (conf): Minimum likelihood that the consequent (RHS)
occurs given the antecedent (LHS), here 80%.
# Summary of rules ----
# Provides an overview of the rules, including their count and statistical
properties like support and confidence.
summary(rules)
# Inspect the top 20 rules ----
# View the 20 rules with the highest confidence.
inspect(sort(rules, by = "confidence")[1:20])
# Sorting by confidence ranks rules by the strength of their predictiveness.
# Visualize the association rules with parallel coordinates plot
# Parallel coordinates plot shows relationships between variables in the
rules.
plot(rules, method = "paracoord", control = list(reorder =
TRUE, alpha = 0.8))
# Variables (LHS and RHS) appear as vertical axes, and connecting lines
represent rules.
# Transparency (alpha = 0.8) reduces clutter for better visualization.
```

B

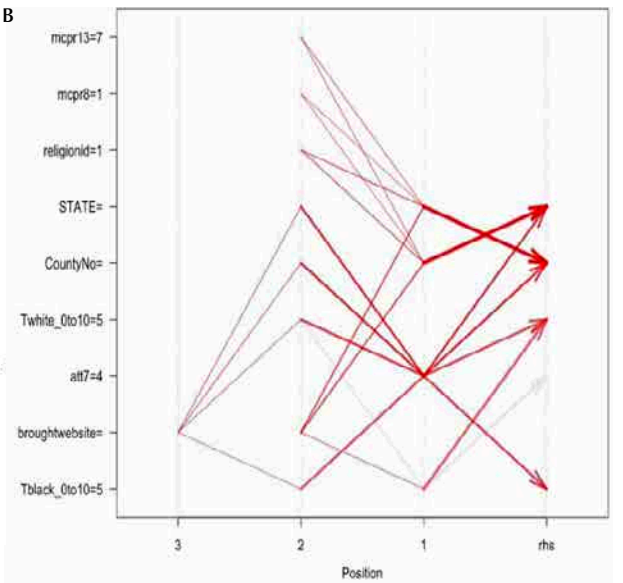

FIGURE 8. Market basket analysis: R code and parallel coordinates plot. Panel A provides the R code for applying the a priori algorithm to derive association rules from transaction data, using a minimum support of 10% and a confidence threshold of 80%. Panel B visualizes the top 9 rules with a parallel coordinates plot. The vertical axes represent the variables involved in the rules, with positions numbered 3, 2, and 1 indicating the sequence from antecedents (LHS) on the left to consequents (RHS) on the right. Connecting lines depict individual rules, where line thickness and color intensity signify confidence levels: Thick, dark red lines indicate high-confidence associations, while gray lines represent weaker relationships. Panel A: Applying the A Priori Algorithm for Market Basket Analysis; Panel B: Parallel Coordinates Plot of Association Rules in Market Basket Analysis.

products are frequently bought together. That is, "transaction" format represents the data as a table where each row is a transaction and each column is an item, with items indicating whether an item is present in the transaction (1 for "Yes," 0 for "No"). For example, in our research, each respondent's set of survey answers can be a transaction. If we have responses like "Political Identity: Conservative" and "IAT Pro-White Bias: High," these responses are treated as items within the transaction (e.g., a participant will get a 1 in the Political Identity: Conservative column if that is true of the participant). Variables without specific levels are transformed into binary items to indicate their presence or absence.

Interpreting Market Basket Analysis. Results from applying MBA to the Project Implicit data set show many expected associations; for example, there is a strong association between "CountyNo" and "State," as demonstrated by rules like "{CountyNo=} => {State=}" with 100% confidence. Understandably, knowing the county number allows us to predict the state with complete certainty and vice versa. Interestingly, we also find associations among variables such as att7 (a 7-point Likert scale measuring racial attitudes), Tblack_0to10 (an 11-point

thermometer rating of warmth toward Black people), and Twhite_0to10 (an 11-point thermometer rating of warmth toward White people). For instance, the rule {att7=4, Tblack_0to10=5} => {Twhite_0to10=5} with a confidence of 91.6% suggests that individuals with neutral racial attitudes on one measure tend to have neutral racial attitudes on the others. Already, we can see how such methods are helpful in revealing, for example, that the zero-point in explicit attitude measures is meaningful and robust across indicators.

Variables like {broughtwebsite=} and {STATE=} reflect whether participants provided any response rather than what the participants responded with. For example, the rule {broughtwebsite=} => {STATE=} shows that participants who responded (with any response) to broughtwebsite were likely to have also answered the STATE question. This could reflect the survey design, where participants reaching broughtwebsite had already answered STATE, or it may suggest that particularly engaged participants tend to respond to all demographic questions. While this rule may not have much theoretical meaning in the current data, the general principle is that researchers can use these rule structures to identify dependencies or predictions between items in their data.

The high lift items observed in many rules, often greater than 3, suggest that the antecedents and consequents appear together much more frequently than would be expected by chance, indicating strong and meaningful associations (Figure 8b). For example, we found that participants who give neutral responses on one measure tend to do so on others, suggesting "no preference" may be a consistent trait. In addition, rules like "{att7=4, Tblack_0to10=5} => {Twhite_0to10=5}" with 91.6% confidence indicate that specific attitudes toward Black individuals are mirrored in feelings toward White individuals. Again, these insights can help design interventions by understanding the relationships among respondents' attitudes and demographics.

CONCLUSION

UML provides a complementary toolkit that can provide new insights beyond traditional statistical approaches already commonly used in social cognition scholarship. Of course, UML is not a replacement for these traditional statistical methods that focus on inference (identifying causal relationships) or prediction (such as forecasting future outcomes). These approaches remain essential for advancing research, but they also require researchers to have a priori expectations about the relationships between variables in their data set. In contrast, UML techniques allow psychologists to explore their data and uncover bottom-up patterns that conventional methods might have missed. Moreover, the outputs of UML techniques—such as the assignment of participants to different clusters—can be used as inputs to later formal statistics or supervised machine learning (Alashwal et al., 2019).

Looking toward the future, the growing volume (i.e., the number of participants) and complexity (i.e., the number of variables) of psychological data sets will benefit from UML tools. Here, we illustrated four UML tools that enable data mining and discovery: clustering (K-means and DBSCAN) that reveals relationships

among variables or participants, and dimension reduction (PCA and MBA) that helps wrangle unwieldy data sets into interpretable and analyzable structures. We hope that by introducing these basic tools of clustering and dimension reduction, researchers will be equipped with the essential concepts underlying UML and also be empowered to engage with increasingly sophisticated applications of UML. Indeed, by laying out the logic and implementation of tools such as PCA and K-means (which are commonly used in psychology), we hope to show that the same basic logic and implementation underwrites seemingly advanced UML tools (e.g., large language models applied to text data or recommendation algorithms applied to social media behavior; e.g., Brady et al., 2023; Charlesworth et al., 2024).

Using UML throughout social cognition will help ensure that our scholarship remains applied and relevant even to complex real-world data. Insights from social cognition have already been used to answer pressing societal challenges ranging from improving intergroup attitudes (Paluck & Green, 2009) to reducing online misinformation (Ecker et al., 2022), and to resolving political conflict (Voelkel et al., 2023). To scale up these insights and show their implications for real-world settings using naturalistic data, we will be best equipped by adding UML to our toolkits.

Funding statement. This project was supported by the National Science Foundation Graduate Research Fellowship and Northwestern University Institute for Policy Research Graduate Research Assistantship, both awarded to Jonathan Doriscar.

Data availability. All data files and code are provided openly at https://github.com/jedoriscar/Unsupervised-Machine-Learning; https://osf.io/52qxl/.

REFERENCES

Alashwal, H., El Halaby, M., Crouse, J. J., Abdalla, A., & Moustafa, A. A. (2019). The application of unsupervised clustering methods to Alzheimer's disease. *Frontiers in Computational Neuroscience*, *13*, 31. https://doi.org/10.3389/fncom.2019.00031

Brady, W. J., Jackson, J. C., Lindström, B., & Crockett, M. J. (2023). Algorithm-mediated social learning in online social networks. *Trends in Cognitive Sciences*, *27*(10), 947–960. https://doi.org/10.1016/j.tics.2023.06.008

Charlesworth, T. E. S., & Banaji, M. R. (2022). Patterns of implicit and explicit attitudes: IV. Change and stability from 2007 to 2020. *Psychological Science*, *33*(9), 1347–1371. https://doi.org/10.1177/09567976221084257

Charlesworth, T. E. S., Ghate, K., Caliskan, A., & Banaji, M. R. (2024). Extracting intersectional stereotypes from embeddings: Developing and validating the Flexible Intersectional Stereotype Extraction procedure. *PNAS Nexus*, *3*(3), pgae089. https://doi.org/10.1093/pnasnexus/pgae089

Charlesworth, T. E. S., Yang, V., Mann, T. C., Kurdi, B., & Banaji, M. R. (2021). Gender stereotypes in natural language: Word embeddings show robust consistency across child and adult language corpora of more than 65 million words. *Psychological Science*, *32*(2), 218–240. https://doi.org/10.1177/0956797620963619

Davani, A. M., Atari, M., Kennedy, B., & Dehghani, M. (2023). Hate speech classifiers

learn normative social stereotypes. *Transactions of the Association for Computational Linguistics*, *11*, 300–319. https://doi.org/10.1162/tacl_a_00550

Dovidio, J. F., & Gaertner, S. L. (2004). Aversive racism. *Advances in Experimental Social Psychology, 36*, 1–52. https://doi.org/10.1016/S0065-2601(04)36001-6

Ecker, U. K. H., Lewandowsky, S., Cook, J., Schmid, P., Fazio, L. K., Brashier, N., Kendeou, P., Vraga, E. K., & Amazeen, M. A. (2022). The psychological drivers of misinformation belief and its resistance to correction. *Nature Reviews Psychology*, *1*(1), 13–29. https://doi.org/10.1038/s44159-021-00006-y

Fan, T., Guo, N., & Ren, Y. (2021). Consumer clusters detection with geo-tagged social network data using DBSCAN algorithm: A case study of the Pearl River Delta in China. *GeoJournal*, *86*(1), 317–337. https://doi.org/10.1007/s10708-019-10072-8

Fiske, S. (1993). Social cognition and social perception. *Annual Review of Psychology, 44*, 155–194. https://doi.org/10.1146/annurev.ps.44.020193.001103

Gaertner, S. L., & Dovidio, J. F. (2000). The aversive form of racism. In C. Stangor (Ed.), *Stereotypes and prejudice: Essential readings* (pp. 289–304). Psychology Press.

Gruijters, S. L. K. (2020). *Using principal component analysis to validate psychological scales: Bad statistical habits we should have broken yesterday II*. https://doi.org/10.31234/osf.io/qbe6k

Hatzenbuehler, M. L., McKetta, S., Goldberg, N., Sheldon, A., Friedman, S. R., Cooper, H. L. F., Beane, S., Williams, L. D., Tempalski, B., Smith, J. C., Ibragimov, U., Mermin, J., & Stall, R. (2020). Trends in state policy support for sexual minorities and HIV-related outcomes among men who have sex with men in the United States, 2008–2014. *JAIDS Journal of Acquired Immune Deficiency Syndromes*, *85*(1), 39. https://doi.org/10.1097/QAI.0000000000002395

Hehman, E., Ofosu, E. K., & Calanchini, J. (2021). Using environmental features to maximize prediction of regional intergroup bias. *Social Psychological and Personality Science, 12*(2), 156–164. https://doi.org/10.1177/1948550620909775

Higgins, E. T., & Bargh, J. A. (1987). Social cognition and social perception. *Annual Review of Psychology, 38,* 369–425. https://doi.org/10.1146/annurev.ps.38.020187.002101

Kodinariya, T., & Makwana, P. (2013). Review on determining of cluster in K-means clustering. *International Journal of Advance Research in Computer Science and Management Studies*, *1*, 90–95.

Kosinski, M. (2019). Computational psychology. In E. J. Finkel & R. F. Baumeister (Eds.), *Advanced social psychology: The state of the science* (2nd ed., pp. 499–524). Oxford University Press.

Malik, A., & Tuckfield, B. (2019). *Applied unsupervised learning with R*. Packt Publishing.

Martinez, J. E. (2024). Analytic racecraft: Race-based averages create illusory group differences in perceptions of racism. *Journal of Experimental Psychology: General, 153*(12), 3042–3061. https://doi.org/10.1037/xge0001673

Mustakim, M. Z. F., Mustafa, A. A., & Rohayati. (2021). Clustering of public opinion on natural disasters in Indonesia using DBSCAN and K-Medoids algorithms. *Journal of Physics: Conference Series*, *1783*(1), 012016. https://doi.org/10.1088/1742-6596/1783/1/012016

Nosek, B. A., Smyth, F. L., Hansen, J. J., Devos, T., Lindner, N. M., Ranganath, K. A., Smith, C. T., Olson, K. R., Chugh, D., Greenwald, A. G., & Banaji, M. R. (2007). Pervasiveness and correlates of implicit attitudes and stereotypes. *European Review of Social Psychology*, *18*(1), 36–88. https://doi.org/10.1080/10463280701489053

Paluck, E. L., & Green, D. P. (2009). Prejudice reduction: What works? A Review and assessment of research and practice. *Annual Review of Psychology*, *60*(1), 339–367. https://doi.org/10.1146/annurev.psych.60.110707.163607

Paluck, E. L., Porat, R., Clark, C. S., & Green, D. P. (2021). Prejudice reduction: Progress and challenges. *Annual Review of Psychology, 72*, 533–560. https://doi.org/10.1146/annurev-psych-071620-030619

Pargent, F., Schoedel, R., & Stachl, C. (2023). Best practices in supervised machine learning: A tutorial for psychologists. *Advances in Methods and Practices*

in Psychological Science, 6(3), Article 25152459231162559. https://doi.org/10.1177/25152459231162559

Pearson, A. R., Dovidio, J. F., & Gaertner, S. L. (2009). The nature of contemporary prejudice: Insights from aversive racism. *Social and Personality Psychology Compass*, *3*(3), 314–338. https://doi.org/10.1111/j.1751-9004.2009.00183.x

Roy, R., II. (2021, December 29). Auto-clustering with auto-elbow. *Medium*. https://bobrupakroy.medium.com/auto-clustering-32c28b3946f1

Settles, I. H., & Buchanan, N. T. (2014). Multiple groups, multiple identities, and intersectionality. In V. Benet-Martínez & Y.-Y. Hong (Eds.), *The Oxford handbook of multicultural identity* (pp. 160–180). Oxford University Press.

Vavpotič, D., Knavs, K., & Cvelbar, L. K. (2021). Using a market basket analysis in tourism studies. *Tourism Economics*, *27*(8), 1801–1819. https://doi.org/10.1177/1354816620944264

Vitriol, J. A., & Banaji, M. R. (2024). Eliciting cognitive consistency increases acceptance of implicit bias. *Journal of Experimental Psychology. General*, *153*(11), 2859–2878. https://doi.org/10.1037/xge0001596

Vitriol, J. A., & Moskowitz, G. B. (2021). Reducing defensive responding to implicit bias feedback: On the role of perceived moral threat and efficacy to change. *Journal of Experimental Social Psychology*, *96*, 104165. https://doi.org/10.1016/j.jesp.2021.104165

Voelkel, J. G., Chu, J., Stagnaro, M. N., Mernyk, J. S., Redekopp, C., Pink, S. L., Druckman, J. N., Rand, D. G., & Willer, R. (2023). Interventions reducing affective polarization do not necessarily improve anti-democratic attitudes. *Nature Human Behaviour*, *7*(1), 55–64. https://doi.org/10.1038/s41562-022-01466-9

Xu, K., Lofaro, N., Nosek, B. A., Greenwald, A. G., Axt, J., Simon, L., Frost, N., & O'Shea, B. (2024, February 1). *Race IAT 2002-2023*. Retrieved from osf.io/52qxl

Yarkoni, T., & Westfall, J. (2017). Choosing prediction over explanation in psychology: Lessons from machine learning. *Perspectives on Psychological Science*, *12*(6), 1100–1122. https://doi.org/10.1177/1745691617693393