Segmentation & Dimensionality Reduction

Business Analytics — Lecture 8

Assoc. Prof. Nguyen Binh Minh

Agenda

- Why segmentation and DR in business
- k-means: intuition → algorithm → diagnostics
- GMM & EM: soft clustering
- PCA for compression, noise removal, visualization
- RFM framework for customer segmentation
- Exercises

Learning Objectives

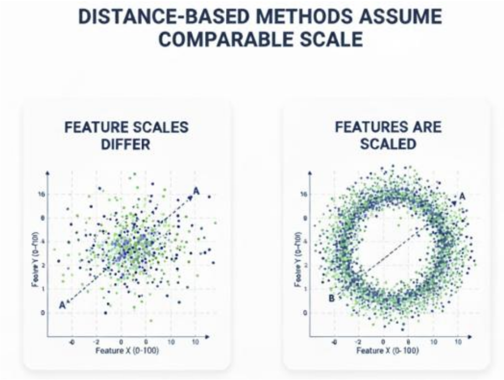
- Explain when/why to use clustering and PCA
- Implement and interpret k-means and GMM outputs
- Select k/components using Elbow, Silhouette, BIC/AIC
- Use PCA to reduce dimensionality and visualize segments
- Apply RFM to derive actionable customer segments

Why Segmentation?

- Personalization, targeted marketing, pricing
- Resource prioritization for sales/service
- Discover structure in high-dim data (product, user)

Data Preparation

- Feature engineering: domain-informed metrics
- Scaling/standardization is critical (z-score, min-max)
- Handling outliers, missing values



Distance Metrics

- **Euclidean (default for k-means)**
  - The "straight-line" or "ruler" distance between two points. It measures pure magnitude.
  - Default for: K-Means clustering.
  - Key Consideration: Highly sensitive to feature scale. Standardization (e.g., StandardScaler) is almost always required.
- **Cosine for direction/similarity in sparse data**
  - Measures the angle (direction) between two vectors, ignoring their magnitude.
  - High-dimensional, sparse data like text analysis (TF-IDF) or recommender systems.
  - Key Consideration: Use when the orientation of data points is more important than their absolute values.
- **Mahalanobis (accounts for covariance)**
  - A statistical distance that measures how many standard deviations a point is from the center (mean) of a distribution.
  - Best for: Outlier detection and clustering data where features are correlated.
  - Key Consideration: It automatically accounts for the covariance matrix of the data, making it scale-invariant.

k-means: Intuition

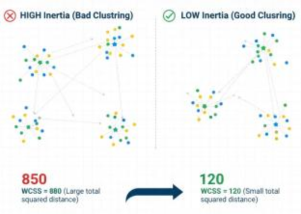
- Goal: to partition n data points into k distinct, non-overlapping clusters.
- Core Idea:
  - Each cluster is represented by its centroid (the mean or "center" of all points in that cluster).
  - Each data point is assigned to the cluster with the nearest centroid.
- The Process (Iterative):
  1. Randomly place k centroids.
  2. Assign: Assign each point to its closest centroid (forms k clusters).
  3. Update: Recalculate the centroid (mean) for each new cluster.
  4. Repeat: Repeat steps 2 and 3 until the centroids stop moving (convergence).

k-means Objective

- Primary Goal: to find the set of k centroids that minimizes the Within-Cluster Sum of Squares (WCSS).
- This metric is also called Inertia or Sum of Squared Errors (SSE).
- What is WCSS?
  - It is the total sum of the squared Euclidean distances from every single data point to the centroid of its assigned cluster.

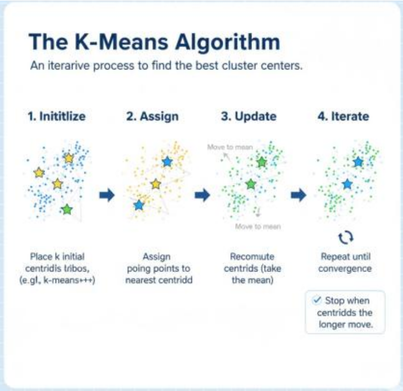
$$WCSS = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2$$

- The Intuition
  - We are trying to make the clusters as "tight" or "compact" as possible.
  - A low WCSS (Inertia) means all points are very close to their respective cluster centers.
- Statistical Equivalence: Minimizing the WCSS is mathematically equivalent to minimizing the variance within each cluster. The "Update" step of K-Means (moving the centroid to the mean) is precisely the action that minimizes this value for that cluster.



k-means Algorithm

- Init k centroids (random or k-means++)
  - Assign points to nearest centroid
  - Recompute centroids; iterate until convergence



Initialization Strategies

- **Random vs. k-means++Random**
  - Initialization:
    - How: Simply picks k random data points from your dataset to be the initial centroids.
    - Pro: Very fast.
    - Con: Can be "unlucky." It might pick multiple centroids in the same dense region, leading to a poor cluster and getting stuck in a bad local minimum.
  - k-means++ (The Smart Default):
    - How: A probabilistic method designed to get a better spread.
    - Step 1: Pick the first centroid randomly.
    - Step 2: For every other point, calculate its distance to the nearest already-chosen centroid.
    - Step 3: Pick the next centroid, where points far away from existing centroids have a higher probability of being chosen.
    - Pro: Vastly improves the quality of the final clustering and speeds up convergence.
- **The Local Minima Problem**
  - The Problem: K-Means is a "greedy" algorithm ➔ get "stuck" in a local minimum (a "pretty good" solution) ➔ fail to find the global minimum (the best possible solution).
  - The Solution: Multiple Restarts (n_init)
    - Run the entire algorithm multiple times (e.g., n_init=10) using different random starting seeds
    - The best one: the lowest WCSS (Inertia).

Scaling & Outliers

- Feature Scaling
  - Why? K-Means uses Euclidean distance. Features with large scales (e.g., Income) will dominate features with small scales (e.g., Age).
  - Solution: Always use StandardScaler (mean=0, std=1) so all features contribute equally.
- Handling Outliers
  - Why? Centroids are based on the mean. A single outlier will "drag" the centroid, skewing the entire cluster.
  - Solutions:
    - Trimming: Remove extreme outliers before clustering.
    - RobustScaler: Use a scaler (like RobustScaler) that is not sensitive to outliers.

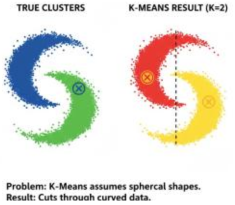
Distance Choices in Practice

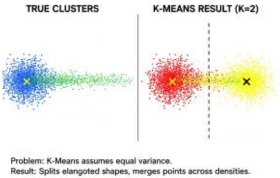
- Cosine K-Means
  - When to use: Ideal for high-dimensional, sparse data (e.g., text documents, TF-IDF vectors).
  - Why? It ignores magnitude (e.g., document length) and clusters based on direction or similarity (e.g., topic content).
  - The Goal: Groups vectors that point in a similar direction.

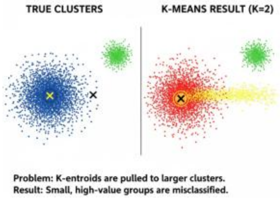
- K-Prototypes
  - When to use: Your dataset has a mix of numeric and categorical features (e.g., "Age" and "City").
  - Why? K-Means (Euclidean) only handles numbers, and K-Modes only handles categories.
  - The Goal: K-Prototypes combines both:
    - It uses Euclidean distance for numeric features.
    - It uses Hamming distance (mismatch count) for categorical features.

K-Means Failure Modes

- **Non-Convex Shapes**
  - Issue: Algorithm assumes spherical clusters.
  - Result: Fails to detect complex geometries (e.g., "crescent" or ring shapes).
- **Varying Densities**
  - Issue: Euclidean distance does not account for variance (spread).
  - Result: Sparse clusters are often split or merged into dense noise.
- **Unequal Cluster Sizes (Imbalance)**
  - Issue: Centroids are pulled toward the larger group.
  - Result: Smaller, high-value segments are often misclassified.

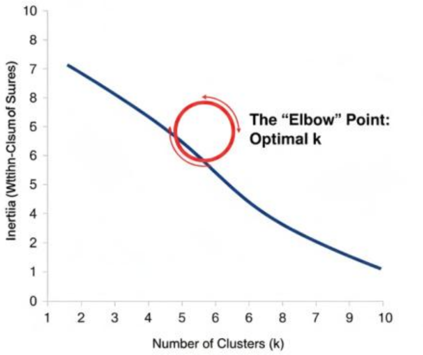






Choosing k: Elbow

- The Concept
  - Plots Inertia (Within-Cluster Sum of Squares) against the number of clusters (k).
  - Inertia measures how tightly grouped the data points are within a cluster.
- The Visual Cue
  - Look for the "Elbow" point: The specific value of k where the curve bends and the rate of decrease slows down significantly.
- Business Interpretation
  - Diminishing Returns: Beyond the elbow, adding more clusters (complexity) yields minimal improvement in model accuracy.



Problem: Choosing k for K-Means. Solution: Look for the point of diminnhing returns.

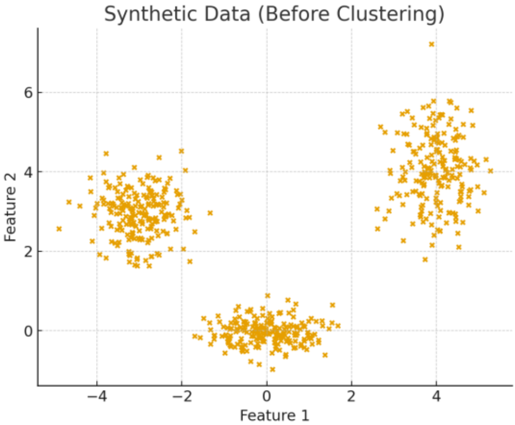
Choosing k: Silhouette

- The Concept
  - Measures how similar a point is to its own cluster (Cohesion) compared to other clusters (Separation).
  - Used to validate the consistency within clusters.
- Interpreting the Score (-1 to +1)
  - Close to +1: Well-clustered (Dense & clearly separated).
  - Close to 0: Overlapping clusters (on the boundary).
  - Negative (< 0): Data point is likely placed in the wrong cluster.
- Business Value
  - Provides a precise metric when the "Elbow" is ambiguous.
  - Ensures segments are distinct enough to justify different strategies (e.g., distinct marketing campaigns).

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

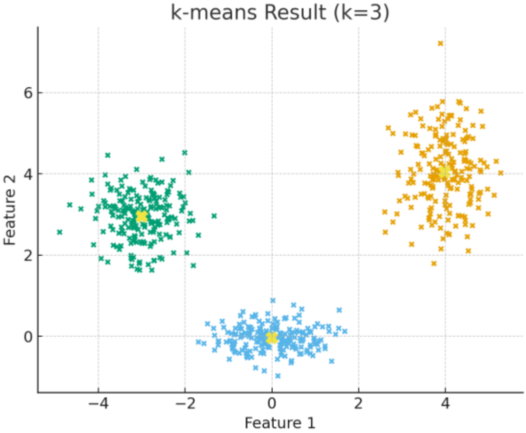
Data Before Clustering

- Synthetic 2D features; three underlying groups



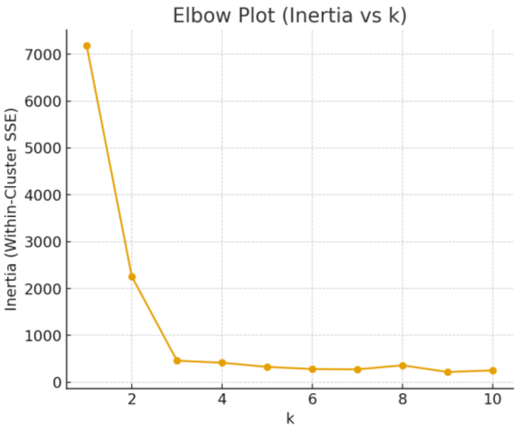
Demo: k-means Result (k=3)

- Clusters & centroids shown



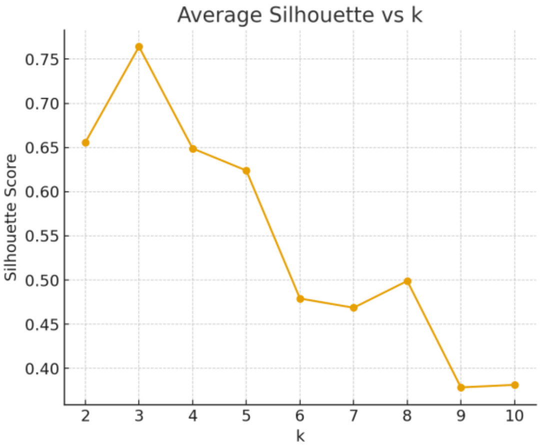
Elbow Plot

- Inertia decreases with k; elbow around ground-truth



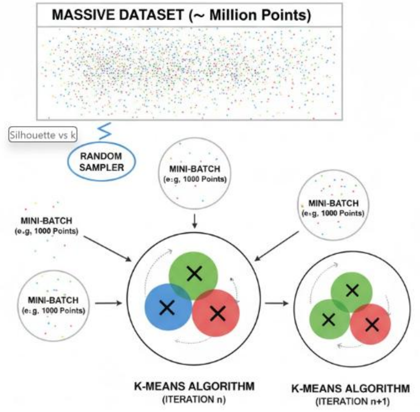
Silhouette vs k

- Peak indicates better k (not guaranteed)



Mini-batch k-means: Scaling to Big Data

- The Problem with Standard K-Means
  - Requires the entire dataset to be in memory for every iteration.
  - Extremely slow and computationally expensive for massive datasets.
- The Mini-Batch Solution
  - Random Sampling: Updates centroids using small, random subsets (batches) of data at each step.
  - Incremental Learning: The model "learns" and adjusts centroids gradually, batch by batch.
- Performance Trade-off
  - Speed: Converges much faster (often 2x–100x faster).
  - Accuracy: Result is an approximation. Inertia is slightly higher than standard K-Means, but the difference is usually negligible for business insights.



Practical Diagnostics: Is the Model Usable?

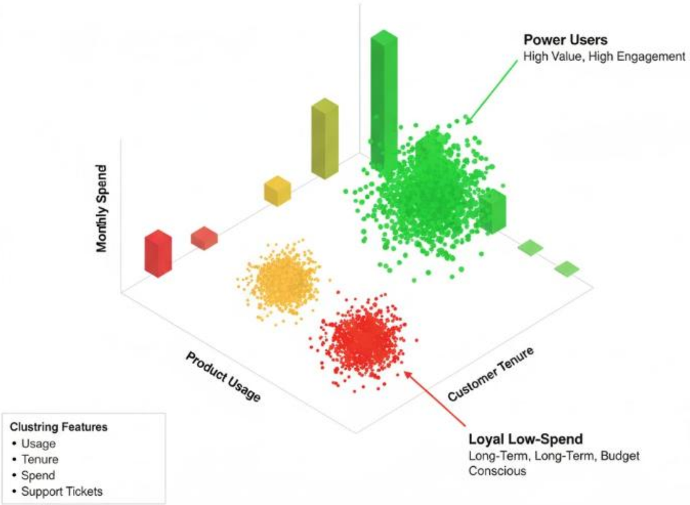
- Stability Check (Random Seeds)
  - Action: Run K-Means multiple times with different random initializations (seeds).
  - Centroid Drift: If centroids shift significantly between runs, the solution is unstable (likely due to noise or incorrect k).
- Cluster Size Distribution
  - Sanity Check: Are the cluster sizes relatively balanced?
  - Outlier Buckets: Watch out for "micro-clusters" (e.g., containing <1% of data). These usually capture outliers/noise rather than a valid market segment.
- Business Profiling (Interpretation)
  - Descriptive Stats: Calculate the Mean/Median of key features for each cluster.
  - Persona Building: Translate numbers into labels (e.g., “Cluster 1 = High Income, Low Frequency”).

Case: Customer Features

- Business Goal: Understand customer groups to tailor marketing and service strategies.
- Key Features for Clustering (RFM-like + Engagement)
  - Usage: How often customers use the product/service (e.g., logins/month).
  - Tenure: How long they have been a customer (e.g., months since signup).
  - Spend: Total revenue generated by the customer (e.g., average monthly spend).
  - Support Tickets: Number of support requests (proxy for issues or engagement).
- Interpreted Segments (Example)
  - Power Users: High Usage, High Spend, Moderate Tenure, Few Support Tickets.
  - Loyal Low-Spend: High Tenure, Low Spend, Moderate Usage, Few Support Tickets.
  - At-Risk: Low Usage, Low Tenure, High Support Tickets, Low Spend (New users struggling or old users churning).

Customer Segmentation with K-Keans

Identified Segments based on Key Features



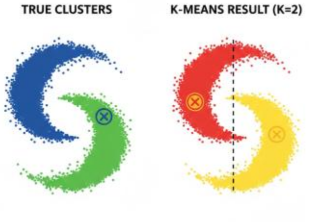
Naming Segments

- Human-Readable Labels for Stakeholders
  - Goal: Replace abstract cluster numbers (e.g., "Cluster 3") with intuitive, business-oriented names.
  - Method: Summarize the core characteristics of each cluster (based on descriptive statistics) into a memorable persona.
  - Example: "High-Value Loyalists," "Newbie Explorers," "Churn Risks.“
- Attach KPIs & Recommended Actions
  - Goal: Make segments actionable by linking them to specific business metrics and strategies.
  - KPIs: Identify key performance indicators relevant to each segment (e.g., Churn Rate, Average Order Value, Engagement Score).
  - Actions: Develop tailored interventions, marketing campaigns, or product features for each segment.
  - Example (for "Churn Risks"): KPI = Reduced churn rate; Action = Proactive support, win-back offers.

Naming and Activation Customer Segments

Beyond K-Means: Gaussian Mixture Models (GMM)

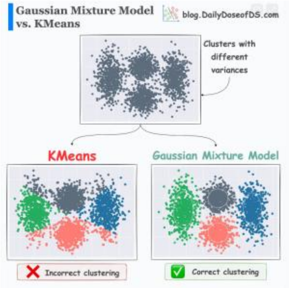
- Overcoming Shape Rigidity (Elliptical Flexibility)
  - Limitation: K-Means assumes clusters are spherical (circles), failing to capture elongated or stretched data patterns.
  - GMM Advantage: Models clusters as ellipses. It explicitly handles variance and covariance, allowing the cluster boundaries to stretch and rotate to fit the actual data distribution.
- From Hard to Soft Clustering
  - Limitation: K-Means forces a "Hard Assignment" (a point belongs 100% to one cluster).
  - GMM Advantage: Provides Soft Membership (Probabilistic assignment).
  - Business Value: Identifies "borderline cases"—customers who sit between segments (e.g., a user who is 60% "Loyal" but 40% "At-Risk"). This nuance allows for more sophisticated targeting.



Problem: K-Means assumes sphercal shapes.
Result: Cuts through curved data.

GMM: Concept

- Data as a Mixture of Gaussians
  - Idea: GMM assumes that your entire dataset is generated from a combination (a "mixture") of several underlying, simpler Gaussian distributions.
  - Analogy: Imagine your customer base isn't one big group, but actually a blend of "segments," each with its own typical behaviors that follow a bell curve.
- Each Gaussian Component Has 3 Key Parameters:
  - Mean μ: The center of the cluster (similar to K-Means centroid).
    - Represents: The typical value of features for that segment.
  - Covariance Σ: The shape and orientation of the cluster (how stretched/rotated it is).
    - Represents: The variance and correlation among features within that segment (e.g., how "Spend" and "Usage" vary together). This allows for elliptical shapes.
  - Weight π: The proportion of data points belonging to this cluster.
    - Represents: How large or dominant this segment is in the overall dataset.



GMM: Covariance Types

1. Spherical (Simple)
   - Shape: Round circles (spheres).
   - Constraint: Variance is equal in all directions.
   - Note: Effectively reduces GMM to K-Means. Least flexible, fastest computation.
2. Diagonal (Axis-Aligned)
   - Shape: Ellipses aligned with the X/Y axes.
   - Constraint: Clusters can stretch, but cannot rotate.
   - Note: Assumes features are independent (uncorrelated).
3. Full (Complex)
   - Shape: Any elliptical shape.
   - Constraint: Clusters can stretch and rotate freely.
   - Note: Most flexible but computationally expensive. Prone to overfitting if data is scarce.
4. Tied (Shared)
   - Shape: All clusters share the same shape and orientation.
   - Note: Useful when you assume segments differ only by location (mean), not by variance.

The EM Algorithm: How GMM Learns

- The "Chicken and Egg" Problem
  - We don't know the cluster parameters μ, Σ, π without knowing which points belong to which cluster.
  - We don't know the points' membership without the parameters.
  - Solution: An iterative loop called EM.
- Step 1: E-Step (Expectation) – "Soft Assignment“
  - Action: Compute Responsibilities.
  - Calculate the probability that each data point belongs to each cluster based on current parameters.
  - Example: Point A is 80% Cluster 1, 20% Cluster 2.
- Step 2: M-Step (Maximization) – "Update Parameters“
  - Action: Re-calculate parameters to fit the new assignments.
  - Update Mean (μ): Move center towards the weighted average of points.
  - Update Covariance (Σ): Stretch/rotate to fit the spread.
  - Update Weight (π): Adjust based on total probability mass.
- Convergence
  - Repeat E & M until the Log-Likelihood (total model fit) stops increasing.

Selecting #Components

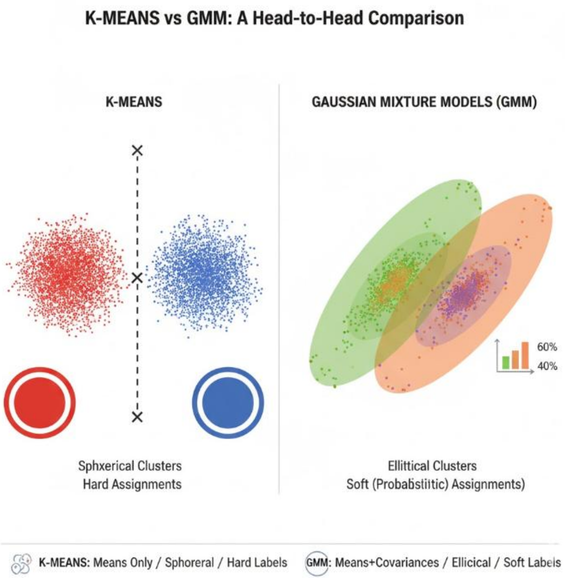
- The Problem: Overfitting
  - Unlike Inertia, Log-Likelihood keeps increasing as you add more components.
  - Result: Without a penalty, the model would eventually create one cluster per data point (perfect fit, zero utility).
- The Solution: Penalize Complexity
  - AIC (Akaike) & BIC (Bayesian): Metrics that balance model fit (Likelihood) against model complexity (Number of Parameters).
  - The Rule: Lower is Better. We look for the minimum point on the curve.
  - BIC vs. AIC: BIC imposes a stricter penalty for complexity. It prefers simpler models, making it generally safer for Business Analytics to avoid overfitting.
- Cross-Validation (Alternative)Split data into Train/Test sets.
  - Evaluate if the Log-Likelihood on the Test set remains high. If it drops while Training score rises ➔ Overfitting.

Soft Assignments: The Power of Probability

- The Concept: "Shades of Grey“
  - Hard Clustering (K-Means): A customer is either in Segment A OR Segment B (0 or 1).
  - Soft Clustering (GMM): A customer has a probability of belonging to each segment (e.g., 70% Segment A, 30% Segment B).
- Strategic Thresholds for Action
  - Core Members: High probability (> 80%).Action: Standard retention/loyalty campaigns.
  - Borderline Cases (Fuzzy): Split probabilities (e.g., 50/50).Insight: These customers are "on the fence" or transitioning between behaviors.
  - Action: Personalized Nudges. They need specific incentives to push them definitively into a high-value segment.
- Example Strategy
  - If P(VIP) > 0.9: Auto-upgrade to Gold Member.
  - If 0.5 < P(VIP) < 0.9: Send "Challenge" (Spend $50 more to unlock Gold).

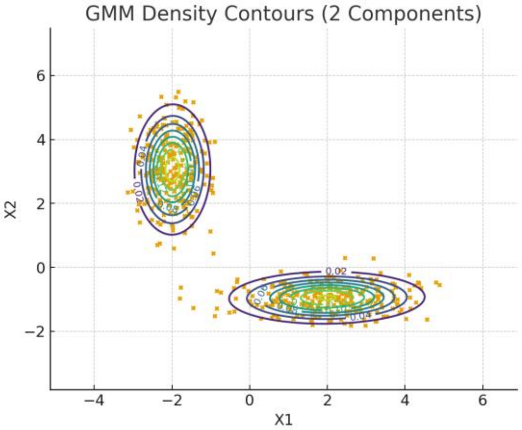
k-means vs GMM

Feature	**K-Means**	**Gaussian Mixture Models (GMM)**
Model Parameters	Only **Means** (Centroids)	**Means (μ), Covariances (Σ), Weights (π)
Cluster Shape	**Spherical** (Circles/Spheres)	**Elliptical** (Flexible shapes, can rotate)
Assignment Type	**Hard Labels** (Each point belongs 100% to one cluster)	**Soft Labels** (Probabilities of belonging to each cluster)
Algorithm	Iterative centroid updates	**EM Algorithm** (E-Step & M-Step)
Choosing k/Components	**Elbow Method** (Inertia)	**AIC/BIC** (Information Criteria)
Handling Density	Struggles with varying densities	Handles varying densities effectively
Computational Cost	Faster, scales well to large N	Slower, more complex (especially with Full Covariance)
Key Advantage	Simplicity, speed	Flexibility, nuanced insights, captures complex data


K-MEANS vs GMM: A Head-to-Head Comparison
K-MEANS
GAUSSIAN MIXTURE MODELS (GMM)
60%
40%
Sphxerical Clusters
Hard Assignments
Ellittical Clusters
Soft (Probabstiitic) Assignments)
K-MEANS: Means Only / Sphoreral / Hard Labels
GMM: Means+Covariances / Ellicical / Soft Labels

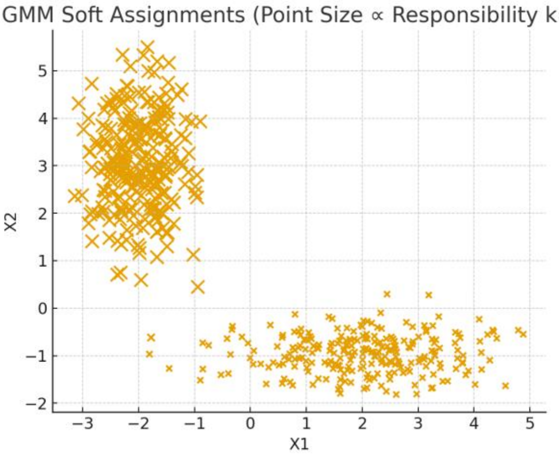
GMM Density Contours (Demo)

- Contours visualize elliptical components



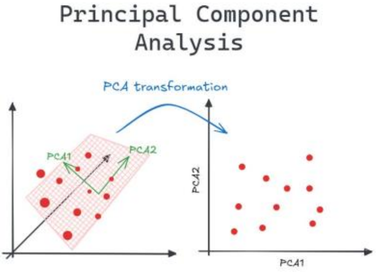
GMM Soft Responsibilities (Demo)

- Point size ∝ responsibility for a component



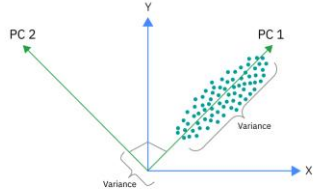
PCA: Why Reduce Dimensionality?

- Noise Reduction
  - Filters out irrelevant variance and random noise, allowing the model to focus on the true signal.
- Data Compression
  - Reduces storage requirements and computational costs by summarizing data into fewer components.
- Visualization
  - Enables human interpretation of complex datasets by projecting high-dimensional data into 2D or 3D plots.
- Curse of Dimensionality
  - Mitigates model performance degradation caused by data sparsity in high-dimensional spaces.



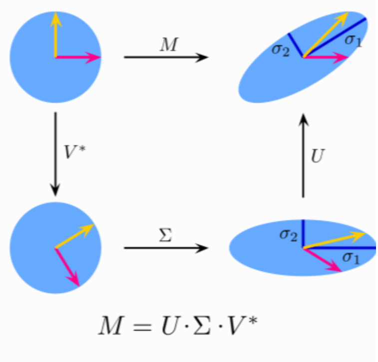
Variance & Covariance

- **Orthogonal Directions:** PCA identifies the specific axes along which the data varies the most. These directions are perpendicular to each other, ensuring independence.
- **Covariance Matrix:** This mathematical structure summarizes the joint variability of your data. PCA essentially "diagonalizes" this matrix to isolate the pure signal from the noise.



PCA via SVD

- **Center Data**
  Subtract the mean from the dataset matrix to ensure zero-centered features:
  $$X \leftarrow X - \mu$$
- **Apply SVD**
  Decompose the centered matrix into three components:
  $$X = U \Sigma V^T$$
- **Identify Components (PCs)**
  The columns of V (right-singular vectors) represent the principal directions.
- **Calculate Scores**
  The projections of data onto the PCs are given by: Scores = U Σ



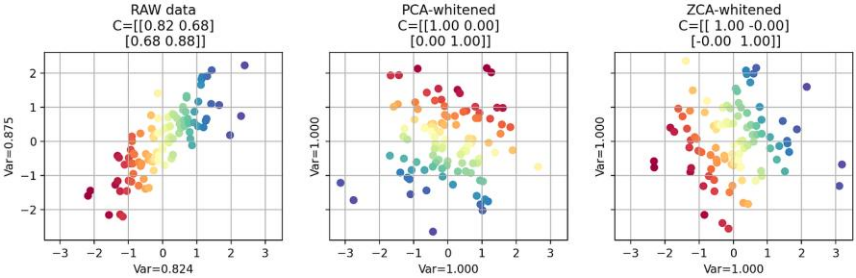
PCA Steps

Step	Action	Formula/Concept
Data Preparation	**Standardize** (optional) and **Center** the data.	$X_{centered} = X - \mu_x$
Compute Components	Find the directions of maximum variance.	**SVD** of the centered data, or **Eigendecomposition** of the Covariance Matrix (Σ).
Select Components	Choose the k components to keep.	Use **Explained Variance** (e.g., 90% cumulative threshold).
Transform Data	Project the original data onto the new k-dimensional subspace.	$Z = X_{centered} \cdot W_k$

Whitening (Optional)

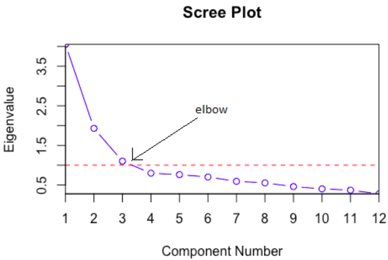
- **Rescale Variance**: Normalizes the Principal Components so that each dimension has unit variance (variance = 1).
- **From Ellipse to Sphere**: Transforms the data distribution from an oriented ellipse into an isotropic sphere.
- **Application**: Essential pre-processing for algorithms assuming isotropic covariance, such as Independent Component Analysis (ICA).

$$Z_{\text{white}} = \frac{\mathrm{PC}_i}{\sqrt{\lambda_i}}$$



Scree Plot

- Visualize Variance:
  - Plots the eigenvalues (variance explained) against the number of principal components.
- The "Elbow" Rule:
  - Identify the point where the curve bends sharply and flattens out.
- Decision Strategy:
  - Keep components before the elbow (signal) and discard those after (noise/scree).



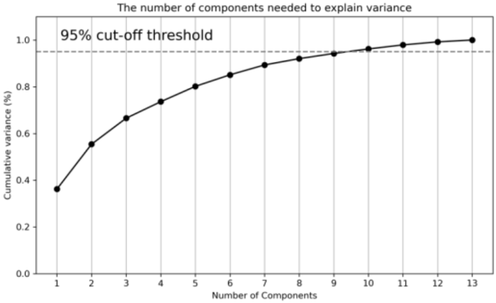
Explained Variance Ratio

$$\frac{\lambda_i}{\sum_{j=1}^{d} \lambda_i}$$

Cumulative Explained Variance

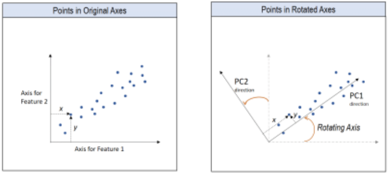
- **Target Threshold:**
  Instead of finding an "elbow," we often select the number of components (m) needed to retain a specific percentage of information (e.g., 90% or 95%).
- **The Goal:**
  Find the smallest m such that the cumulative sum of explained variance meets the requirement.
- **Trade-off:**
  Higher threshold = better reconstruction quality but less compression (higher dimensionality).

$$\sum_{i=1}^{m} VarianceRatio_i$$



PCA Projection (2D)

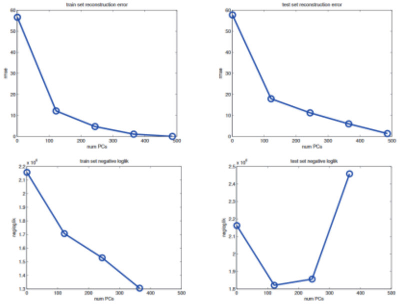
- **Dimensionality Reduction:**
  Data is projected onto the plane defined by the first two Principal Components (PC1 & PC2), which capture the most variance.
- **Visualizing Clusters:**
  Patterns, groupings, and separability between classes often become distinct in this 2D view, even if hidden in higher dimensions.
- **Exploratory Analysis:**
  A critical step for spotting outliers and understanding the intrinsic structure of the dataset before modeling.



Reconstruction Error

- **Definition:**
  It measures the information lost when projecting data onto a lower-dimensional subspace (Mean Squared Error).
- **Inverse Relationship:**
  As you increase the number of components (k), the approximation improves, and the error decreases significantly.
- **Convergence:**
  When k equals the original dimensionality (d), the error becomes zero ($Error = 0$).
- **Approximation Error Formula**

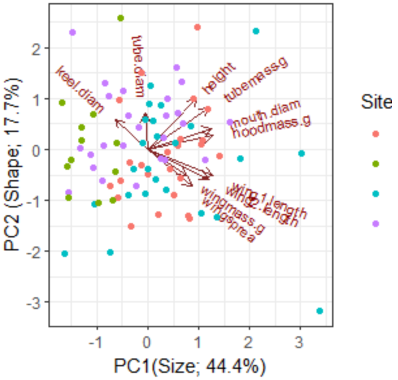
$$\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)}-x_{\text{approx}}\right\|^{2}$$



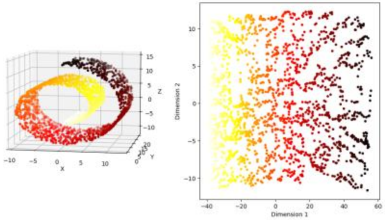
Interpreting Loadings

- **Loadings ϕ**
  Coefficients that define the linear combination. Large absolute values imply the feature strongly influences that component.
- **The Biplot:**
  A powerful dual visualization plotting both *Scores* (samples as dots) and *Loadings* (features as vectors) simultaneously.
- **Reading the Angles:**
  - Angle ≈ 0°: Positive correlation
  - Angle ≈ 180°: Negative correlation
  - Angle ≈ 90°: No correlation (Orthogonal)

$$\mathrm{PC}_1 = \phi_1 X_1 + \phi_2 X_2 + \ldots + \phi_p X_p$$



PCA Caveats

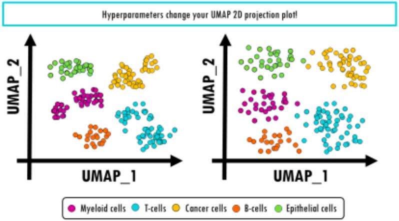


PCA squashes the "Swiss Roll" (non-linear), losing its structure.

- **Linear Limitation:**
  PCA assumes data lies on a linear subspace. It fails to unfold complex, non-linear manifolds (e.g., the "Swiss Roll").
- **Scale Sensitivity:**
  Variables with large magnitudes dominate the variance. *Standardization* is not optional—it's mandatory.
- **Outlier Sensitivity:**
  PCA minimizes least-squares error, meaning extreme outliers can significantly "pull" and distort the principal axes.

Beyond PCA: t-SNE & UMAP

- **Non-Linear Mapping:**
  Techniques like t-SNE and UMAP excel at unfolding complex, non-linear manifolds that PCA (which is linear) flattens and destroys.
- **t-SNE & UMAP:**
  t-SNE is powerful for preserving local clusters. *UMAP* is faster and better balances local vs. global structure.
- **Use with Care:**
  Unlike PCA, these methods are *stochastic* (results vary per run) and highly sensitive to hyperparameters (perplexity, n_neighbors). Axes typically have no interpretable meaning.



RFM Analysis: The Core Concept

- A behavioral segmentation technique used to quantify customer value by analyzing past purchase behavior.

ENGAGEMENT

Recency (R)
How recently did the customer make a purchase?
Measured in **days** since last transaction.

LOYALTY

Frequency (F)
How often does the customer purchase?
Measured in **count** of total transactions.

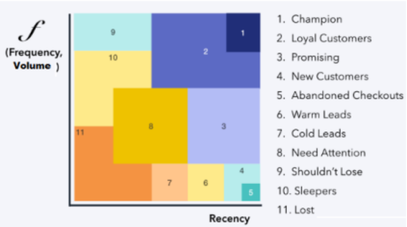
VALUE

Monetary (M)
How much does the customer spend?
Measured in **total revenue** generated.

RFM: Computing Scores

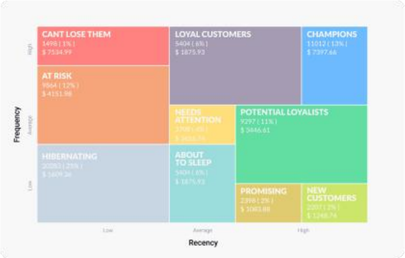
- **Quantile Binning:**
  Divide customers into 5 equal groups (quintiles) for each metric. This normalizes the data into a 1-5 scale.
- **Scoring Logic:**
  F & M: Higher value = Higher score (5 is best).
  - **Recency:** Lower value (fewer days ago) = Higher score (5 is best).
- **Concatenation:**
  Combine the three digits to form a unique segment code.

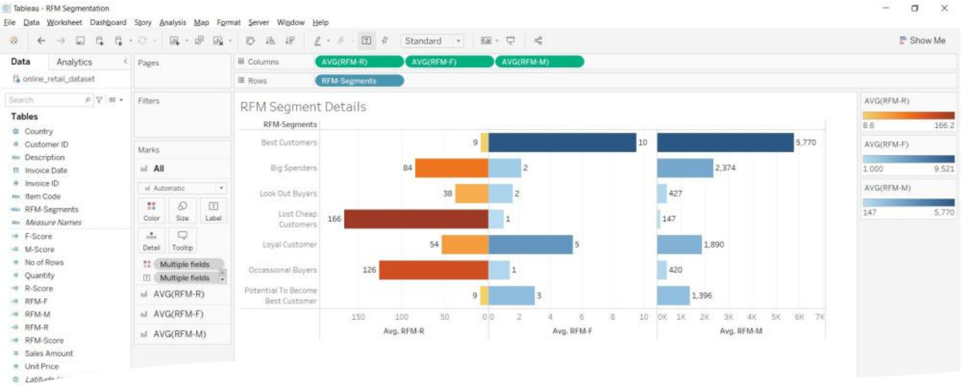
Customer Segments



RFM: Segment Taxonomy

- **Growth & High Value**
  - **Champions:** Bought recently, buy often, and spend the most.
  - **Loyal Customers:** Buy on a regular basis. Responsive to promotions.
  - **Potential Loyalist:** Recent customers with average frequency.
- **Risk & Churn**
  - **At Risk:** Big spenders who haven't purchased lately.
  - **Hibernating:** Last purchase was long ago, low spenders.
  - **Lost:** Lowest recency, frequency, and monetary scores.



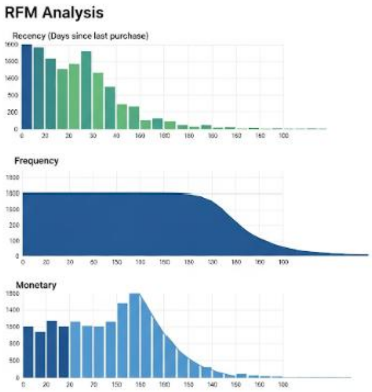


RFM: Dashboard Sketch

- **KPIs per Segment:**
  Monitor critical metrics like *Average Order Value (AOV)*, *Churn Rate*, and *Profitability* specific to each cluster (e.g., Champions vs. At Risk).
- **Trendlines & Migration:**
  Visualize how customers move between segments over time. Are "Potential Loyalists" graduating to "Champions"?
- **Conversion Tracking:**
  Measure the effectiveness of targeted campaigns. Track response rates and ROI for actions directed at specific segments.

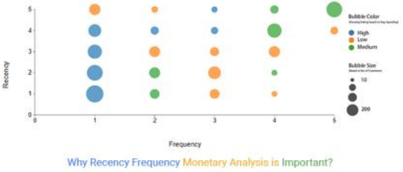
RFM Distributions

- **Skewness is Normal:**
  Real-world customer data rarely follows a bell curve. Expect heavy *right-skewed* distributions (Long Tail).
- **Frequency & Monetary:**
  Typically follow the *Pareto Principle (80/20 Rule)*: A vast majority of customers make few, small purchases, while a few "whales" drive value.
- **Implication for Binning:**
  Standard equal-width bins fail here. *Quantile binning* is essential to ensure segments are balanced and meaningful.



RFM Scatter

- **The X-Y Plane:**
  Plotting *Recency* (X-axis) against *Monetary* (Y-axis) immediately reveals the "Active Spenders" vs. "Lost Cheap" customers.
- **The Third Dimension:**
  We use **Bubble Size** to represent *Frequency*. Larger bubbles indicate customers who transact more often.
- **Identifying Champions:**
  Look for large bubbles in the "Recent & High Spend" quadrant. These are your most valuable, frequent, and engaged users.



$$\text{Bubble Size} \propto \text{Frequency}$$

Pipeline: Hybrid Segmentation

- A robust machine learning workflow combining the interpretability of RFM with the power of PCA and K-means.
- Standardize → PCA → k-means/GMM → Profile + Name

Standardize

Z-score Scaling: Normalize RFM and additional behavioral features to unit variance. Essential for distance-based algorithms.

PCA Projection

Dimensionality Reduction: Extract orthogonal components to handle multicollinearity (e.g., F vs. M) and reduce noise.

Clustering

k-means / GMM: Apply clustering algorithms on the top Principal Components to identify compact, separated groups.

Profile & Name

Interpretation: Inverse-transform cluster centroids back to the original scale to assign personas (e.g., "Whales").

Model Selection & Validation

- **Time-Based Splits:**
  For customer data, random splits leak information. Use *Time-Series Splits* (e.g., Train: Jan-Jun, Test: Jul) to validate stability against seasonality.
- **Monitor Drift:**
  Customer behaviors evolve. Check for *Concept Drift* (e.g., distinct clusters merging over time) to ensure the model remains relevant.
- **Re-train Cadence:**
  Establish a schedule (e.g., Monthly or Quarterly) to refresh the PCA transformation and cluster centroids.

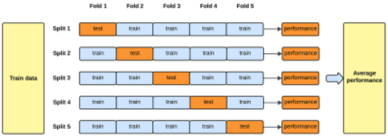


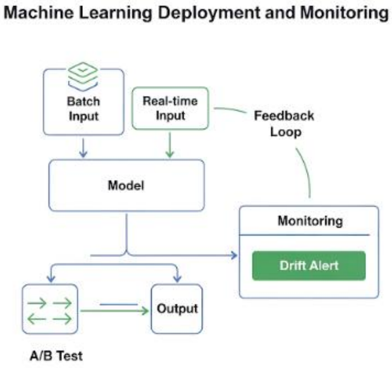
Figure: Expanding Window Strategy ensuring the model is tested on "future" unseen data.

Ethics & Privacy

- **Avoid Sensitive Attributes:**
  Never use protected characteristics (race, gender, religion) as features. Beware of *proxy variables* (e.g., zip codes) that implicitly correlate with them.
- **Explainability (XAI):**
  Stakeholders must understand *why* a customer is profiled. Avoid "black box" models for sensitive decisions (e.g., credit limits) to build trust.
- **Consent & Transparency:**
  Adhere to GDPR/CCPA. Customers have the right to know they are being segmented and must consent to how their data is used.

Deployment & Monitoring

- **Scoring Strategy:**
  - *Batch:* Large-scale, periodic updates (e.g., nightly email lists).
  - *Real-time:* Instant classification via API for live personalization.
- **Validation in Prod:**
  Use *Back-testing* on historical data to estimate lift, followed by live *A/B Testing* (Control vs. Segmented) to measure actual business impact.
- **Drift Alerts:**
  Set automated triggers for data distribution shifts. If the input data changes significantly (Drift), trigger a model re-train.



References & Further Reading

- Hastie, Tibshirani, Friedman — ESL
- Aggarwal — Data Clustering
- Marketing analytics texts on RFM

Summary & Q&A

- k-means vs GMM; PCA trade-offs; RFM practicality
- Selecting k/components; profiling is key