JPMC Take-Home: Census Income Classification & Segmentation

Vishal Gajavelly

February 19, 2026

1 Introduction

This project delivers:

1. a **classifier** to predict whether an individual's income is $\geq \$50K$, and
2. an **unsupervised segmentation** of the population into actionable personas.

The dataset is survey-based and includes a sampling weight. The report emphasizes decisions that make results reliable in practice: handling structural missingness, class imbalance, and building interpretable segments suitable for marketing actions. Beyond model accuracy, the end goal is **decision support**: who to target, how to tailor messaging, and how to allocate limited campaign budget.

2 Dataset

The dataset has 199,523 records and 42 original columns with mixed numeric and categorical features. Two special columns are:

- `weight`: survey sampling weight (used for training/evaluation, not a predictive feature),
- `label`: used to define the binary target.

2.1 Target construction and imbalance

We define `target`=1 for income $\geq \$50K$ and `target`=0 otherwise. Class proportions are $\approx 93.8\%$ negative and $\approx 6.2\%$ positive. This imbalance matters operationally: a naive model can achieve high accuracy by predicting `0` for almost everyone. Therefore, evaluation focuses on PR-AUC and threshold tradeoffs (precision/recall) rather than accuracy alone.

3 Data preparation

Because this is survey/microdata, identical rows may represent different individuals; duplicates were not removed. True missingness was limited (e.g., `hispanic origin`); missing categorical values were imputed as `Unknown`.

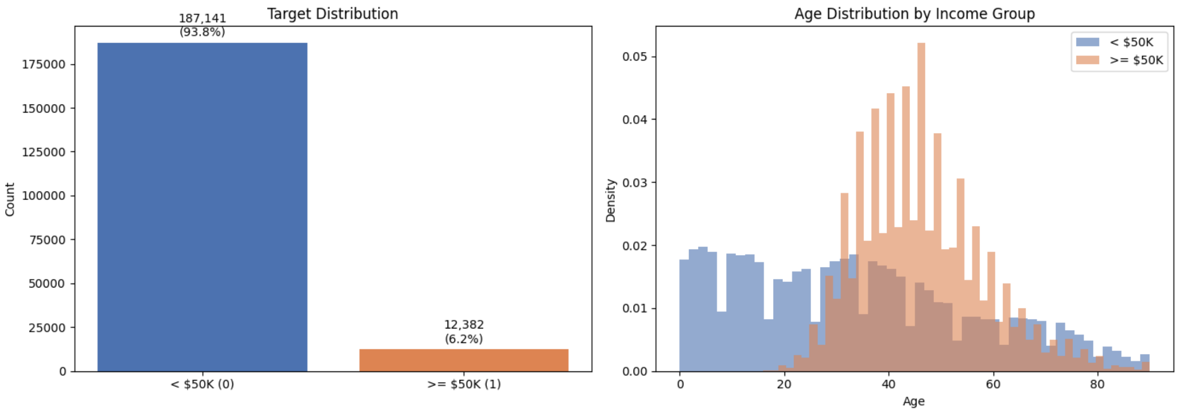

Figure 1: Target distribution and age distribution by income group.

Structural vs unknown missingness. Several categorical fields contain sentinel tokens like `"?"` and `"Not in universe*"`. These were separated to preserve meaning:

- `"?"` → `"Unknown"` (information missing),
- `"Not in universe*"` → `"Not applicable"` (structural non-applicability).

This distinction is important for both supervised learning and segmentation: `Not applicable` often captures real structure (e.g., out of labor force), while `Unknown` is data quality noise.

Numeric sentinel/top-coding. Some numeric maxima (e.g., `wage per hour`=9999, `capital gains`=99999) behave like top-codes/sentinels. These were converted to missing and handled via median imputation inside pipelines to avoid artificial spikes.

4 Feature engineering

Feature engineering aimed to improve signal while staying interpretable: (i) log transforms for skewed financial variables, (ii) bins for nonlinear effects (age/work intensity), and (iii) domain flags (married, full-time, investment indicator). This increased the feature set to 59 columns.

5 Objective 1: Income classification

5.1 Business framing

The classifier supports targeting decisions such as premium outreach or prioritizing limited campaign budget. Because false positives correspond to wasted outreach spend, and false negatives correspond to missed high-value customers, the model should be evaluated in terms of ranking quality and operating tradeoffs:

- **ranking quality**: ROC-AUC and PR-AUC,
- **operating point**: precision/recall/F1 at a chosen threshold.

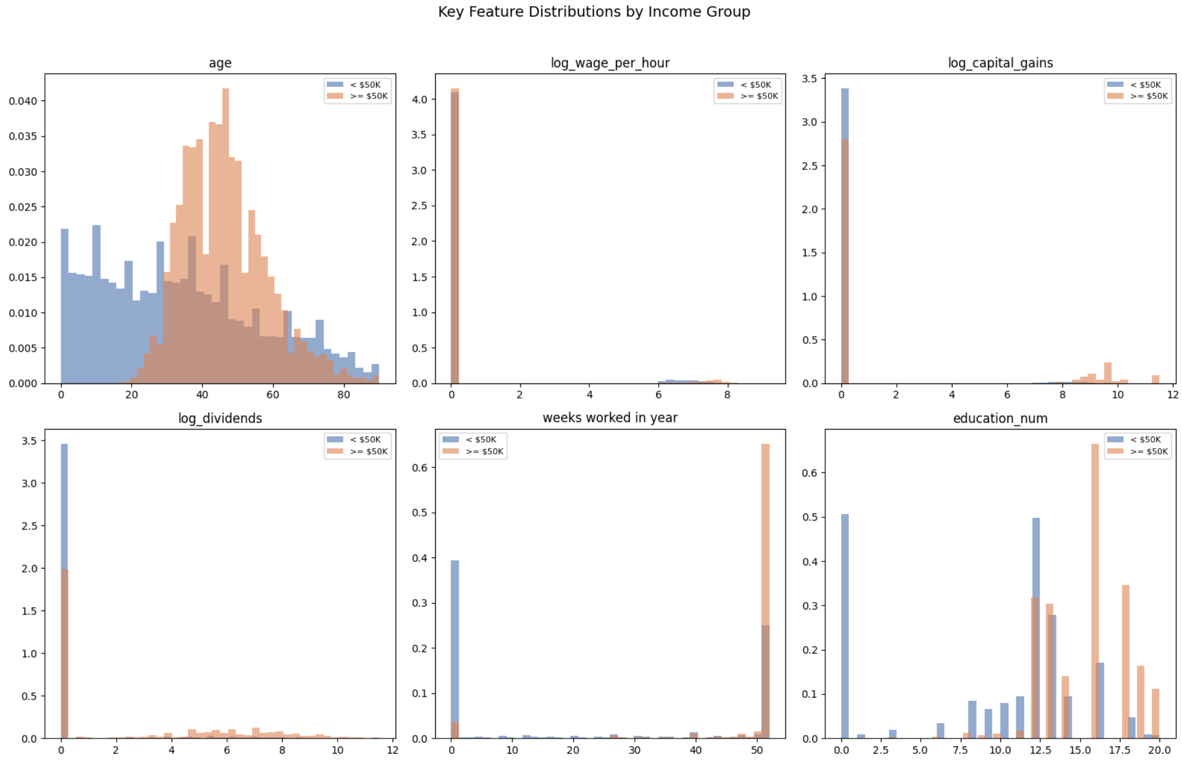

Figure 2: Key numeric distributions (log-transformed where appropriate) by income group.

5.2 Training protocol and weighting

An 80/20 stratified split preserves class proportions. All preprocessing occurs inside **scikit-learn pipelines** to prevent leakage. Survey `weight` is excluded from features and used as **sample_weight** during training and weighted evaluation to approximate population-level performance.

5.3 Models compared and intuition

We compared three baselines with increasing modeling capacity:

- **Logistic Regression**: interpretable linear baseline; strong sanity check under one-hot encoding.
- **Random Forest**: captures nonlinearity and interactions; can under-recall minority class at default threshold.
- **XGBoost**: boosted trees; strong on mixed-type tabular data and generally best for PR-AUC under imbalance.

5.4 Model Selection

XGBoost achieved the best ranking performance under the evaluation protocol. A light randomized hyperparameter search was tested but did not materially improve PR-AUC under the fixed search budget, so the baseline configuration was selected for simplicity and reproducibility.

Table 1: Model comparison (weighted evaluation on test set; threshold = 0.50).

Model	ROC-AUC	PR-AUC	Precision	Recall	F1
Logistic Regression	0.9477	0.6301	0.7326	0.4020	0.5192
Random Forest	0.9497	0.6610	0.8168	0.3370	0.4772
XGBoost (baseline)	**0.9565**	**0.6989**	0.7587	**0.4969**	**0.6005**

Confusion matrix for XGBoost Baseline (raw counts). At threshold 0.50, the unweighted confusion matrix on the test set is:

$$\begin{bmatrix} TN & FP \\ FN & TP \end{bmatrix} = \begin{bmatrix} 37025 & 404 \\ 1249 & 1227 \end{bmatrix}$$

This operating point is **precision-oriented** (few false positives). In deployment, the threshold is a business knob: lowering it increases recall (coverage) at the expense of precision.

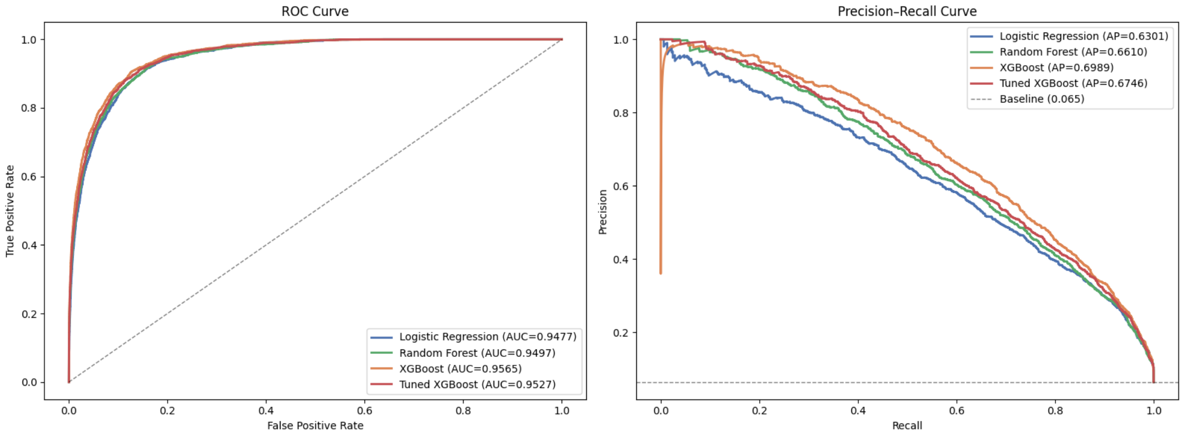

Figure 3: Weighted ROC and PR curves comparing candidate models.

5.5 Objective 1 conclusion

Baseline XGBoost provides strong ranking quality and a practical precision/recall tradeoff for targeting. Threshold selection should be tuned to campaign costs and capacity (e.g., conservative threshold for expensive outreach; lower threshold for broad, low-cost campaigns).

6 Objective 2: Segmentation

The second goal of this project is to produce **meaningful population segments** that can support downstream business use cases like targeting, messaging personalization, and resource allocation. In many marketing and risk settings, a single global strategy is suboptimal because the population is heterogeneous. Segmentation allows:

- **Targeting**: focus campaigns on high-value segments (higher income propensity, stable employment).

- **Tailored messaging**: different segments respond to different product framing (e.g., credit-building vs. investment products).
- **Measurement and monitoring**: track segment size shifts over time and evaluate interventions within comparable groups.

6.1 Feature preparation

Clustering requires an embedding where distance has meaning. Since the dataset contains many categorical variables, I used a standard two-stage approach:

1. **Encode + scale features**:
   - Numeric: median imputation + standard scaling (z-score).
   - Categorical: impute missing as `Unknown` + one-hot encoding.
2. **Reduce dimensionality**: After one-hot encoding, the feature space becomes very high-dimensional and sparse. Distance-based clustering in such spaces often suffers from the *curse of dimensionality*. Therefore, I used **TruncatedSVD** to obtain a compact representation while preserving most of the variance.

Why TruncatedSVD (instead of PCA)? Standard PCA requires dense matrices, but one-hot encoded data is sparse. TruncatedSVD works directly on sparse inputs, making it a practical analogue of PCA for sparse high-dimensional data.

6.2 Choosing the number of SVD components

I explored cumulative explained variance across increasing SVD components and selected a practical cap of 50 components for clustering. This achieved strong compression while retaining most information.

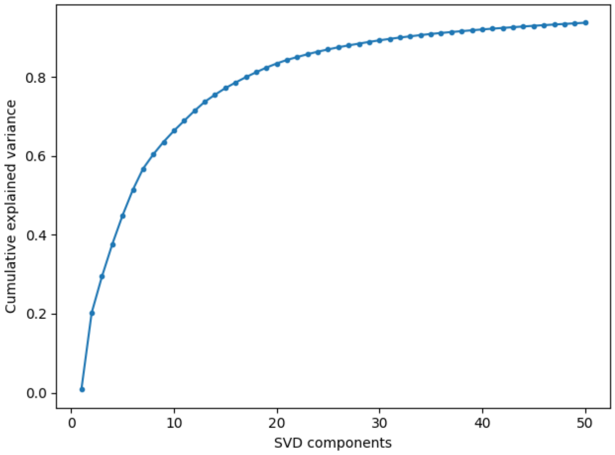

Figure 4: Cumulative explained variance from TruncatedSVD. Using 50 components retains ≈ 93.6% of variance while producing a compact representation suitable for clustering.

6.3 Algorithm selection: KMeans vs GMM

I compared several clustering families on a 15,000-record subsample for efficiency:

- **KMeans**: fast, scalable, good baseline; assumes spherical clusters in embedding space.
- **Gaussian Mixture Models (GMM)**: soft clustering; can model ellipsoidal clusters but is more sensitive and slower at scale.
- **Agglomerative (Ward)**: interpretable hierarchy but computationally heavy for large n and less practical for full data.

The comparison used three standard internal metrics:

- **Silhouette** (higher is better): separation vs compactness.
- **Calinski–Harabasz** (higher is better): ratio of between- vs within-cluster dispersion.
- **Davies–Bouldin** (lower is better): average similarity between each cluster and its most similar cluster.

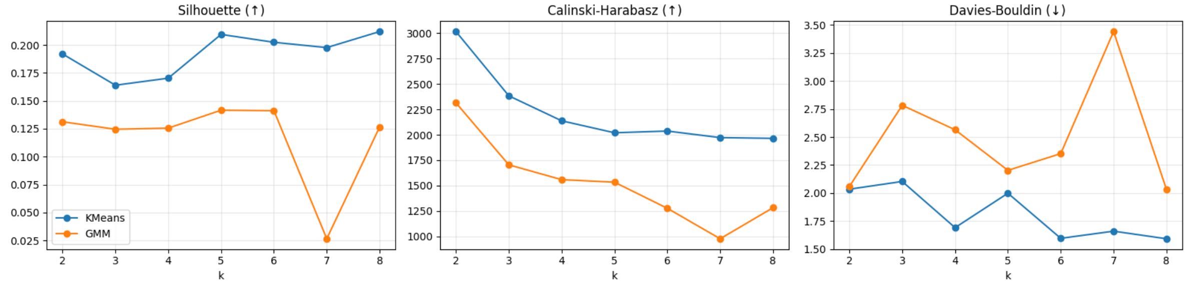

Figure 5: Internal metric comparison across $k = 2..8$ for KMeans and GMM (subsample). KMeans performed consistently better across metrics for this dataset and representation.

Decision: KMeans. Given the scale ($\approx 200k$ rows), stability, and interpretability needs, **KMeans** was selected as the final clustering algorithm. While GMM can be more flexible, the empirical metrics and practical scalability favored KMeans.

6.4 Selecting the number of clusters k

I used a combination of:

- **Elbow curve (inertia)** to detect diminishing returns,
- **Silhouette/CH/DB** to avoid picking k that creates weak or overlapping segments.

Based on metric trends and interpretability, I selected $k = 6$. This value produced segments that were stable and easy to describe as personas.

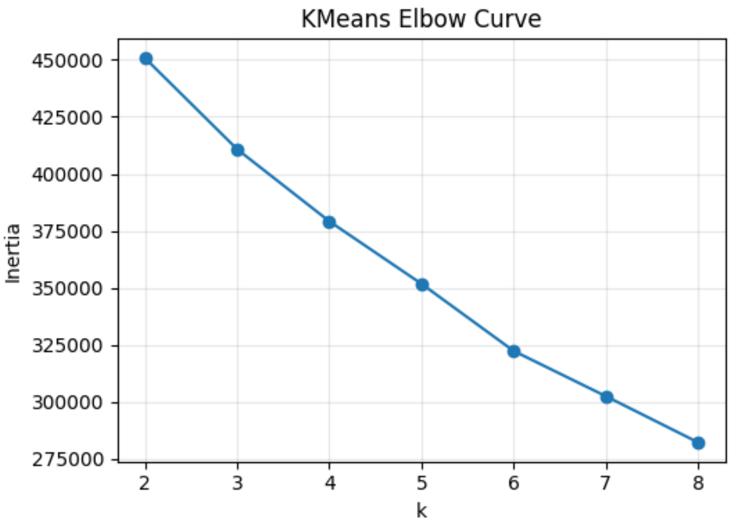

Figure 6: KMeans elbow curve (inertia vs k). The curve shows diminishing improvement beyond a moderate number of clusters, supporting a small number of interpretable segments.

6.5 A key modeling choice: removing a leakage-like categorical field for segmentation

During profiling, I observed that the categorical variable `full or part time employment stat` had a dominant value `Children or Armed Forces` in some clusters. This caused segmentation to over-emphasize administrative status coding rather than broader socio-economic structure.

To avoid a segmentation dominated by one survey-coded category, I dropped this column **for clustering only** (it remains available for profiling after clustering if needed). This led to segments that were more meaningful and less "status-label driven."

6.6 Cluster visualization in reduced space

After fitting KMeans on the full dataset in SVD space (50 components), I plotted a subsample using the first two SVD components for intuition. The plot is not a perfect separation proof (since clustering used 50D space), but it helps confirm that groups occupy different regions and centroids are sensible.

6.7 Stability check (robustness across random seeds)

KMeans can vary with initialization. To ensure the segmentation is not an artifact of random starts, I refit the pipeline across multiple seeds and computed Adjusted Rand Index (ARI) between clusterings.

The segmentation was highly stable:

$$\mathrm{ARI}_{\mathrm{mean}} = 0.936, \quad \mathrm{ARI}_{\mathrm{min}} = 0.839.$$

This indicates the discovered clusters are consistent and reproducible.

6.8 Segment profiling

Because the dataset includes a survey weight (`weight`), cluster summaries were computed using:

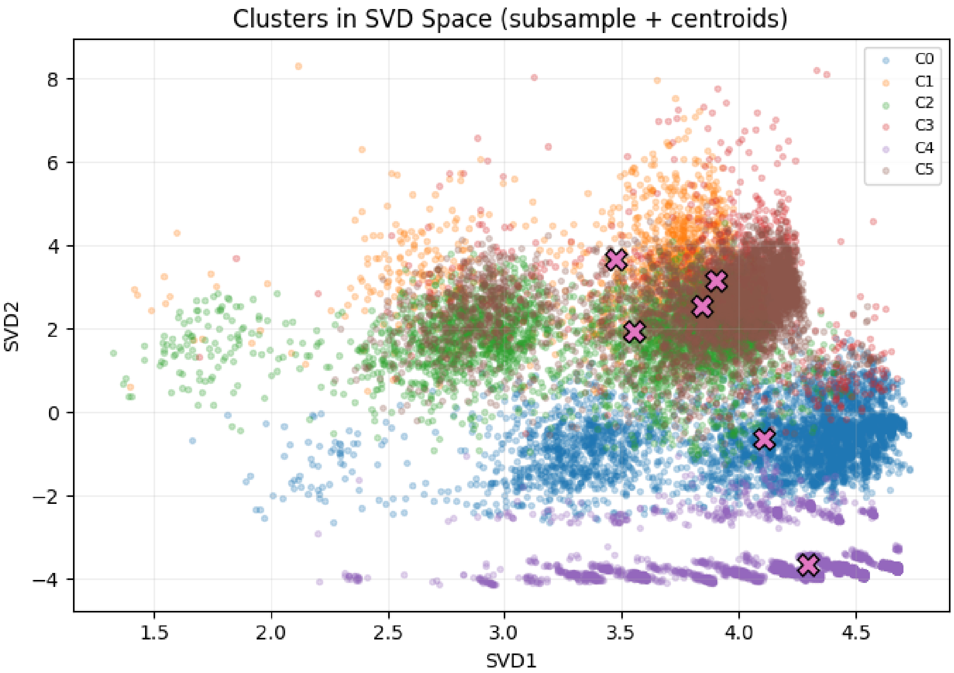

Figure 7: Clusters visualized in SVD space (first two components) on a subsample, with centroids marked. This is a 2D projection of a 50D clustering solution, used mainly for intuition.

- **Unweighted counts** (n) to describe sample size,
- **Weighted share** to estimate population share represented by each segment,
- **Weighted high-income rate** to compare economic value across segments.

To make numeric profiles comparable across features, I created a heatmap of **z-scored weighted means** for key variables (age, education, weeks worked, wage/investment signals, full-time, married).

6.9 Persona identification and segmentation summary

To convert clusters into business-usable segments, I assigned one concise persona name per cluster based on:

- work attachment (weeks worked / full-time),
- investment signals (capital gains/dividends, investment flag),
- age and education,
- weighted high-income propensity.

The final segment names were:

- **C3: Affluent Investors** (small but highest income propensity; strong investment signal)
- **C5: Prime Full-Time Workers** (large; above-average income propensity)

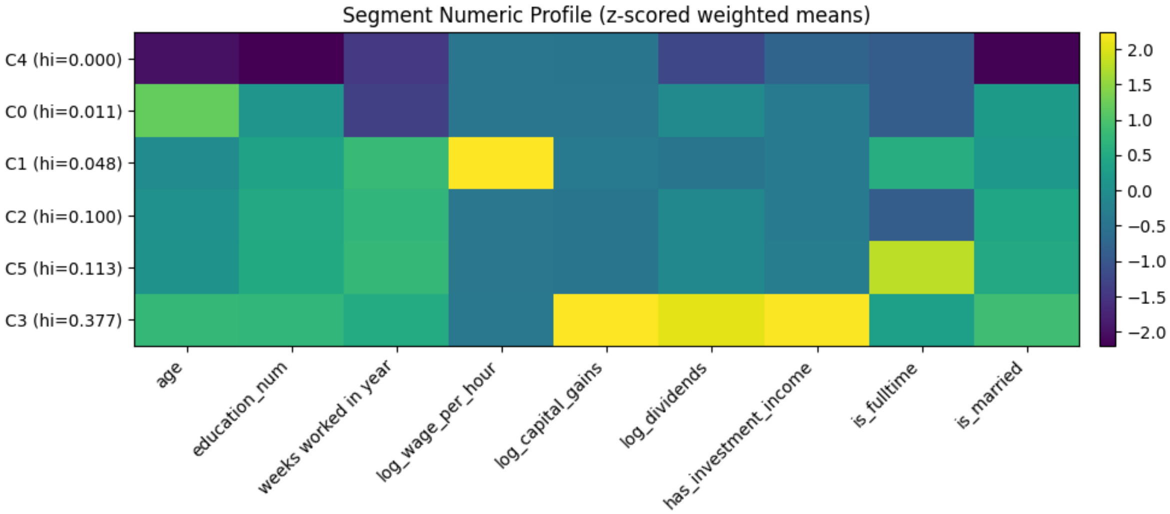

Figure 8: Segment numeric profile heatmap (z-scored weighted means). This highlights which segments are relatively higher/lower on work intensity, investment signals, and demographic attributes.

- **C2: Steady Workers** (large; moderate income propensity)
- **C1: Low-Income Workers** (smaller; below-average income propensity)
- **C0: Older Non-Workers** (large; very low income propensity)
- **C4: Dependents** (large; near-zero income propensity; structurally non-earners)

Why "Dependents" and "Older Non-Workers" are still useful segments. Even though they are not high-income, they are large population groups. In practice, these could correspond to:

- non-income products (basic banking, savings nudges),
- household-level targeting (e.g., family products),
- exclusion lists for certain campaigns.

6.10 Final visualization: segment size vs high-income propensity

To summarize segment business value in one view, I plotted segment size (n) vs weighted probability of earning $\geq \$50K$, with bubble size proportional to weighted population share.

6.11 Tradeoffs and limitations

- **Interpretability vs complexity**: KMeans + SVD gives clear segments, but may miss non-spherical structure. This was an intentional tradeoff for scale and clarity.
- **Projection artifacts**: 2D plots are only for intuition; clustering happens in 50D.
- **Categorical coding effects**: survey-coded categories can dominate segmentation; dropping one problematic field improved persona usefulness.

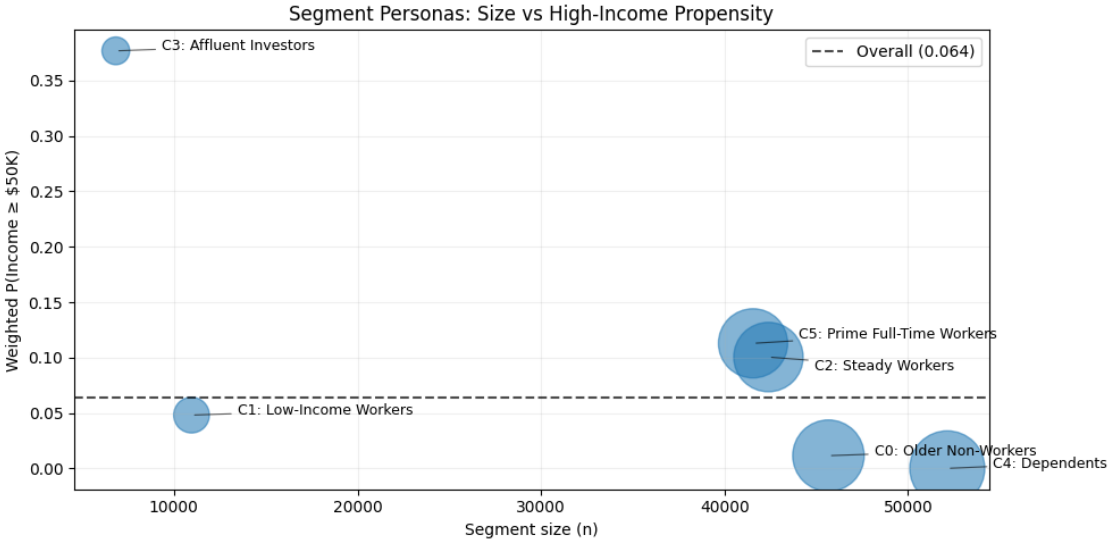

Figure 9: Segment personas: size vs weighted high-income propensity. Bubble size reflects population weight share. The dashed line indicates overall weighted baseline.

- **Unsupervised nature**: clusters are not optimized for predicting the income label; they are optimized for similarity structure. We use income rate only for profiling and business interpretation.

6.12 Conclusion

Objective 2 produced a stable, interpretable segmentation with clear differences in employment attachment, investment behavior, and income propensity. The clustering solution is robust (mean ARI = 0.936) and yields personas that can be directly used for targeting and messaging strategies.