Census Income Classification and Customer Segmentation

Project Report

David Dooling

February 28, 2026

1 Introduction

A retail business client seeks to identify two population groups for targeted marketing: individuals earning less than $50,000 and those earning $50,000 or more. The client has access to 40 demographic and employment variables for any person of interest, along with a labeled dataset from the 1994–1995 U.S. Census Bureau Current Population Surveys (199,523 records).

This report describes the end-to-end pipeline for (1) training a binary classifier that predicts income class, and (2) building an interpretable segmentation model that produces actionable marketing rules.

2 Data Exploration and Preprocessing

2.1 Dataset Overview

The dataset contains 199,523 observations with 40 demographic and employment features, a sampling weight, and a binary income label (`- 50000.` vs. `50000+.`). The class distribution is highly imbalanced: 187,141 records (93.8%) below $50K and 12,382 records (6.2%) at or above $50K, a ratio of approximately 15:1.

Each record carries a *weight* indicating how many people in the general population it represents due to stratified sampling. These weights are incorporated during both training and evaluation to ensure that model performance metrics reflect the true population distribution.

2.2 Feature Types

Of the 40 original variables, we retain 38 predictive features after dropping three non-informative columns: the sampling weight (used separately), year (constant within each survey), and a veteran's administration questionnaire flag.

- **Numeric (11):** age, wage per hour, capital gains, capital losses, dividends from stocks, weeks worked in year, and several recoded indices (industry, occupation, employer count, self-employment, veteran's benefits).
- **Categorical (27):** education, class of worker, marital status, race, sex, tax filer status, household composition, citizenship, migration codes, and others.

2.3 Preprocessing Decisions

1. **No imputation needed.** The dataset has no missing values—all fields are populated.
2. **No one-hot encoding.** LightGBM natively handles categorical features via an optimal split-finding algorithm that considers subsets of categories, avoiding the curse of dimensionality that one-hot encoding introduces for high-cardinality features.
3. **No feature scaling.** Tree-based models are invariant to monotonic transformations of features, so normalization is unnecessary for training.
4. **Stratified train/test split.** An 80/20 split stratified by label preserves the 15:1 class ratio in both partitions (159,618 train / 39,905 test).

3 Model Architecture: Why LightGBM

3.1 Gradient-Boosted Decision Trees for Tabular Data

For structured/tabular data, gradient-boosted decision tree (GBDT) ensembles consistently outperform deep neural networks in both predictive accuracy and training efficiency. Several large-scale benchmarks support this:

- Grinsztajn et al. (2022) evaluated tree-based models against neural networks across 45 tabular datasets and found that *"tree-based models remain the best standard method for medium-sized tabular data."* Neural networks struggled with irregular feature distributions and uninformative features—both common in census data.
- Shwartz-Ziv & Armon (2022) reached similar conclusions, showing that XGBoost and LightGBM outperformed deep learning architectures on most tabular benchmarks while training orders of magnitude faster.
- Borisov et al. (2022) surveyed deep learning on tabular data and noted that despite recent architectures (TabNet, FT-Transformer, SAINT), GBDTs remain dominant on "classical" tabular tasks.

The core advantage of tree ensembles on tabular data is their ability to naturally handle heterogeneous features (mixed numeric and categorical), capture non-linear interactions without manual feature engineering, and remain robust to uninformative features. Neural networks, by contrast, assume spatial or sequential structure in the input and require careful normalization, embedding layers for categoricals, and extensive hyperparameter tuning—often for marginal or no improvement on tabular data.

3.2 LightGBM Specifically

LightGBM (Ke et al., 2017) extends the GBDT framework with two key innovations:

1. **Gradient-based One-Side Sampling (GOSS):** Prioritizes training on instances with large gradients (high prediction error), reducing computation without sacrificing accuracy.
2. **Exclusive Feature Bundling (EFB):** Bundles mutually exclusive sparse features to reduce the effective number of features, accelerating histogram construction.
3. **Native categorical handling:** Instead of one-hot encoding, LightGBM finds optimal category subsets at each split, which is particularly valuable for our 27 categorical features (some with 40+ categories).

These properties make LightGBM an excellent fit for this problem: large dataset (200K rows), mixed feature types, high-cardinality categoricals, and severe class imbalance.

4 Training Algorithm

4.1 Hyperparameter Optimization

We use Optuna (Akiba et al., 2019) with Tree-structured Parzen Estimator (TPE) sampling to search over 14 hyperparameters across 50 trials. Each trial evaluates a parameter configuration using 5-fold stratified cross-validation with AUC-ROC as the optimization metric. Optuna's median pruner terminates unpromising trials early (9 of 50 trials were pruned), reducing total search time.

The search space covers learning rate, tree structure (depth, leaves, min samples), regularization (L1, L2, min split gain), subsampling, class weighting, and categorical smoothing parameters. The best configuration (trial #47, CV AUC 0.9543) uses:

- Learning rate: 0.016, with 2,000 maximum estimators and early stopping at round 50
- Moderate tree complexity: 167 leaves, max depth 7
- Conservative class weighting: `scale_pos_weight` = 2.7
- Strong regularization: `reg_lambda` = 4.42, `min_split_gain` = 0.71

4.2 Final Model Training

The final model is trained on the full training set (159,618 rows) with a 5% internal validation split for early stopping. Training halted at iteration 849 (of 2,000 maximum), indicating effective regularization against overfitting.

5 Evaluation

The model is evaluated on the held-out test set (39,905 rows, never seen during training or hyperparameter search).

Table 1: Classification performance on the held-out test set.

Metric	**Value**
AUC-ROC	0.957
AUC-PR	0.709
F1 Score (weighted)	0.635
Precision (50K+)	0.613
Recall (50K+)	0.659

The AUC-ROC of 0.957 indicates excellent discrimination between the two income classes. The moderate F1 score (0.635) reflects the inherent difficulty of the 15:1 class imbalance—the model achieves high recall (66% of high-income individuals correctly identified) while maintaining reasonable precision (61%).

Table 2: Confusion matrix (sample-weighted counts representing population estimates).

	Predicted $<$ 50K	**Predicted $\geq$ 50K**
Actual $<$ 50K	63,083,064	1,870,037
Actual $\geq$ 50K	1,527,610	2,955,870

When weighted by population, the model correctly classifies 95% of the population, with a false positive rate of just 2.9% on the majority class.

6 Feature Importance and Explainability with SHAP

6.1 Why SHAP

Understanding *why* a model makes specific predictions is critical for business adoption. SHAP (SHapley Additive exPlanations; Lundberg & Lee, 2017) is grounded in cooperative game theory and provides several guarantees that ad-hoc importance measures lack:

- **Local accuracy:** SHAP values for a single prediction sum to the difference between the prediction and the baseline.
- **Consistency:** If a feature's contribution increases in a new model, its SHAP value never decreases.
- **Missingness:** Features not present receive zero attribution.

For tree ensembles, `TreeExplainer` computes exact SHAP values in polynomial time, making it practical even for large datasets.

6.2 Split-Based vs. SHAP Importance

LightGBM's default importance metric counts how often each feature is used for splits. This favors high-cardinality features (many possible split points) over genuinely impactful ones. SHAP importance (mean |SHAP value|) measures each feature's actual contribution to predictions. The two rankings diverge substantially:

Table 3: Top 10 features: split-based vs. SHAP importance.

Rank	Split-Based	Rank	SHAP
1	age	1	age
2	detailed industry recode	2	tax filer stat
3	detailed occupation recode	3	education
4	dividends from stocks	4	weeks worked in year
5	num persons worked for employer	5	major occupation code
6	capital gains	6	sex
7	weeks worked in year	7	detailed household & family stat
8	capital losses	8	num persons worked for employer
9	wage per hour	9	dividends from stocks
10	live in this house 1 year ago	10	major industry code

Notably, "tax filer stat" and "education"—two features with high business interpretability—rank #2 and #3 by SHAP but are absent from the split-based top 10. The high-cardinality "detailed industry/occupation recode" features rank highly by split count but contribute far less to actual predictions. We use the SHAP ordering for all downstream visualizations and segmentation.

Figure 1: Global SHAP summary (beeswarm) plot. Each dot is one test observation; horizontal position shows the SHAP value (impact on prediction), color shows the feature value (red = high, blue = low). Age, tax filer status, education, and weeks worked are the dominant predictors.

7 Segmentation Model

7.1 Approach

While the LightGBM classifier achieves strong predictive performance, its ensemble of 849 trees is not directly interpretable. For the client's marketing team, we need simple, actionable rules.

We train a single **depth-4 decision tree** (max 16 leaf segments) on the top 10 SHAP features

using the training data with sample weights. The shallow depth ensures that every rule can be expressed as a chain of at most 4 conditions—concise enough for a non-technical stakeholder to apply.

7.2 Parallel Coordinates Visualization

Before extracting rules, we visualize how the two income classes separate across the top features using parallel coordinates plots. Each line represents one individual; lines are colored by income class (blue for $<$\$50K, orange for $\geq$\$50K).

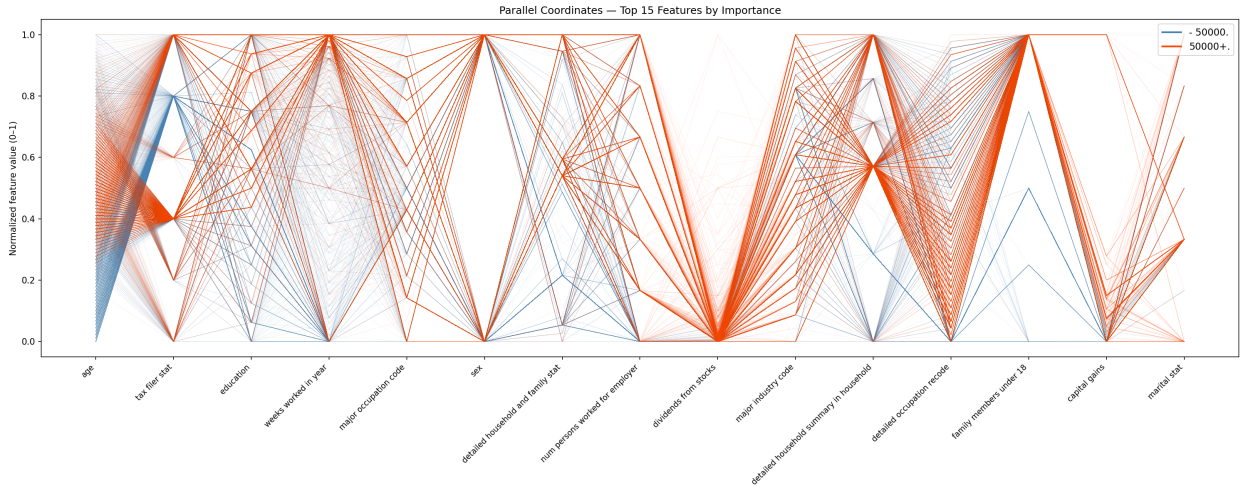

Figure 2: Parallel coordinates plot of the top 15 SHAP features (test set, 3,000 samples per class). Columns are ordered left-to-right by decreasing SHAP importance. Clear separation is visible on age, weeks worked, and dividends from stocks.

An interactive version of this visualization is provided as a standalone HTML file (`artifacts/parallel_coordinat` built with Facebook's HiPlot library, allowing the client to dynamically filter, reorder columns, and brush selections.

7.3 Decision Tree Segmentation Rules

The depth-4 decision tree produces 16 leaf segments. The key high-income segments are:

Figure 3: Visual decision tree (depth 4, 16 leaves). Orange nodes lean toward ≥$50K; blue nodes lean toward <$50K. Node color intensity indicates class purity.

Segment 1 — Full-time employed men over 28 (largest high-income segment):

> Weeks worked > 36 AND age > 28 AND male → weighted population: 75.3M predicted ≥$50K (including 57.5M with no dividend income, and 43.1M with dividend income > 0).

Segment 2 — High dividend earners (regardless of employment):

> Dividends from stocks > $3,992 → predicted ≥$50K (3.3M weighted pop.)

Segment 3 — Young full-time workers with investment income:

> Weeks worked > 36 AND age 25–28 AND dividends > $48 → predicted ≥$50K (0.9M weighted pop.)

The tree achieves 81% overall accuracy on the test set—lower than LightGBM's 95%, but the tradeoff is full interpretability. The rules can be directly implemented in a marketing CRM system without any machine learning infrastructure.

8 Business Recommendations

8.1 Marketing Strategy

Based on the segmentation model, we recommend the following targeting strategy:

1. **Premium product marketing:** Target Segment 1 (full-time employed men over 28)—the largest high-income group by population. This segment responds to career-oriented and family-oriented premium offerings.
2. **Investment and wealth management:** Target Segment 2 (high dividend earners)—this group has demonstrated investment activity and is likely receptive to financial products, luxury goods, and wealth-preservation services.

3. **Aspirational marketing:** Target Segment 3 (young full-time workers with early investment income)—an emerging high-income cohort responsive to growth-oriented financial products, technology, and career development services.
4. **Budget-conscious offerings:** For the below-$50K population (94% of total), emphasize value propositions, affordability, and essential services.

8.2 Deployment Recommendation

For real-time scoring of new individuals, deploy the LightGBM model (serialized as `model.pkl`) behind an API endpoint. For batch marketing campaigns, apply the decision tree rules directly in SQL or the CRM system—no ML infrastructure needed.

8.3 Model Monitoring

The model was trained on 1994–1995 census data. Income distributions, employment patterns, and demographic compositions shift over time. We recommend:

- Retraining on updated census data annually
- Monitoring prediction drift (comparing predicted vs. actual class distributions)
- Re-running SHAP analysis after retraining to verify feature importance stability

9 Future Work

- **Threshold optimization:** The current 0.5 decision threshold could be tuned based on the client's cost of false positives vs. false negatives for each marketing campaign.
- **Fairness audit:** Features like sex, race, and Hispanic origin are among the top predictors. A fairness analysis (equalized odds, demographic parity) should precede any deployment that affects individual outcomes.
- **Feature engineering:** Interaction features (e.g., age $\times$ education, weeks worked $\times$ occupation) may further improve discrimination.
- **Deeper segmentation:** Increasing tree depth to 5–6 would produce finer segments at the cost of interpretability. Alternatively, clustering (e.g., k-prototypes for mixed data types) could complement the supervised segmentation.

10 References

[1] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama. "Optuna: A Next-generation Hyperparameter Optimization Framework." *KDD*, 2019.

[2] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, G. Kasneci. "Deep Neural Networks and Tabular Data: A Survey." *IEEE TNNLS*, 2022.

[3] L. Grinsztajn, E. Oyallon, G. Varoquaux. "Why Do Tree-Based Models Still Outperform Deep Learning on Typical Tabular Data?" *NeurIPS*, 2022.

[4] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree." *NeurIPS*, 2017.

[5] S.M. Lundberg, S.-I. Lee. "A Unified Approach to Interpreting Model Predictions." *NeurIPS*, 2017.

[6] R. Shwartz-Ziv, A. Armon. "Tabular Data: Deep Learning Is Not All You Need." *Information Fusion*, 2022.