ML Exploration Project: Income Classification & User Segmentation

Zekai Su

Columbia University

Project Overview

Marketing efficiency is critical in the financial industry because customer acquisition and retention costs are high, and generic campaigns often waste budget on low-value audiences. A reliable income classification model helps prioritize outreach to higher-income customers (e.g., for premium products or higher-value offers), while a segmentation model helps tailor messaging, channels, and promotions to different customer profiles.

This project aims to complete 2 tasks to serve the above goal by applying machine learning approaches: (1) identifying individuals likely to earn ≥ $50,000, and (2) creating a segmentation model that groups people with similar demographic and employment profiles to support differentiated targeting strategies.

Dataset quick view:

The data provides 40 predictors with 199,523 records. The dataset is weighted census microdata from the 1994 and 1995 Current Population Surveys (CPS). Each record contains demographic and employment variables plus a survey weight indicating how many people in the U.S. population that record represents due to stratified sampling, and an income label: income ≥ $50K vs < $50K.

Methodology Overview

Part 1: Supervised Learning Model for Classification

Goal:

- Predict the binary label using demographic/employment features.

Potential Modeling Method

In this task, I followed the standard supervised learning workflow: Data cleaning -> Baseline model -> Fine-tune baseline model + Other models -> Result evaluation -> Final model report. The baseline model I chose is logistic regression for its interpretability, and I also use an XGBoost classifier for its stronger predictability and non-linear nature.

Evaluation Metrics

The ≥$50K class is a minority, so accuracy is dominated by the majority (<$50K). F1 better reflects the business tradeoff between wasted marketing spend (false positives) and missed high-income prospects (false negatives).

Therefore, I choose the F1 score as the prime metric to evaluate the model, as it balances the precision(avoiding wasting marketing spend) and recall(capturing more high-income groups).

Part 2: Unsupervised Learning Model for Segmentation

Goal:

- Build marketing segments, then describe how segments differ and how marketing can use them.

Potential Modeling Method

Inspired by my past experience, I constructed 2 different approaches for this segmentation task: rule-based segmentation and ML-based segmentation. After constructing the 2 different segmentation methods, I compare the results and interpret the business meaning behind the cluster.

Data Cleaning & Preprocessing

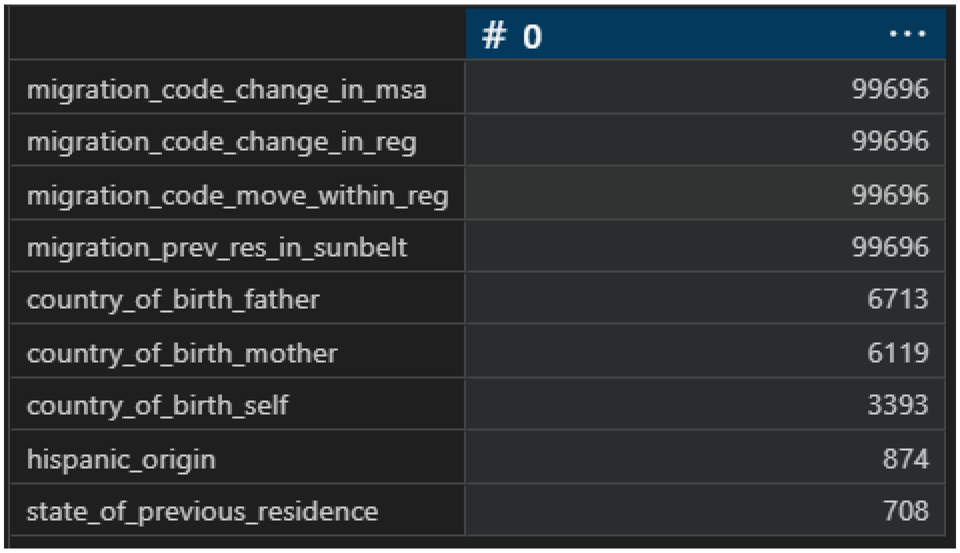
The data cleaning method is the same for both parts of the project.

Step 1: Convert the label into a binary variable:

- label = 1 if income ≥ $50K
- label = 0 otherwise

Step 2: Handling Missing Value:

Note: there is no missing value in the numerical column, but many '?' in categorical variables.



	# 0
migration_code_change_in_msa	99696
migration_code_change_in_reg	99696
migration_code_move_within_reg	99696
migration_prev_res_in_sunbelt	99696
country_of_birth_father	6713
country_of_birth_mother	6119
country_of_birth_self	3393
hispanic_origin	874
state_of_previous_residence	708

(Figure 1: number of '?' in different categorical columns)

To handle the '?' in categorical variables, the approach I take is:

- replaced '?' with NaN
- filled missing categorical values with 'unknown.'

Step 3: Train/Val/Test Split(Part 1)

- The data includes records from the years 1994 and 1995. Instead of doing a random selection in all the data, I decided to use 1994 for train/val and 1995 for test. As this will be closer to the real-world application, we can only use the past year's data to predict the next year.
- Train/Validation: 1994 (80-20 split)
- Final test: 1995

Step 4: Feature Engineering and scaling

- Numerical: Standard scaling (only for logistic regression model)
- Categorical: One-hot encoding
- Exclude year(survey year), label(target), and weight from the training set.

Part 1: Supervised Learning Model for Classification

Baseline: Logistic Regression

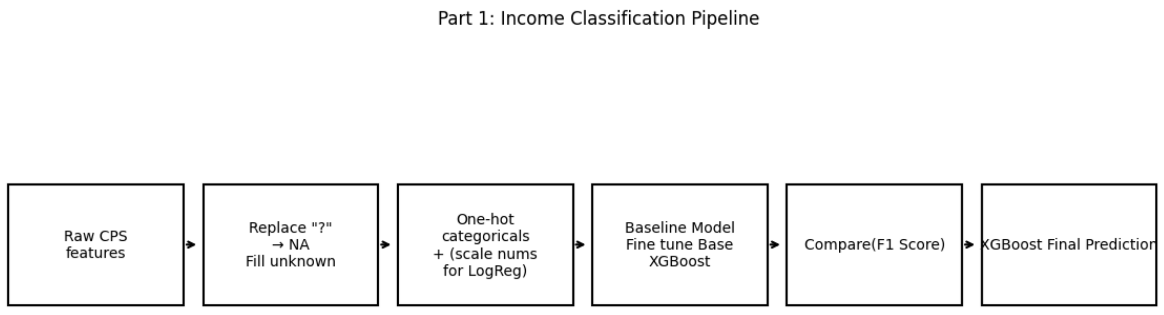
- A weighted Logistic Regression model using one-hot features and evaluated on the weighted validation set. As this is a baseline model, the decision threshold is 0.5
- Baseline threshold tuning improved weighted F1 substantially (optimal threshold ≈ 0.28).

Tuned Logistic Regression

- I used the GridSearchCV over regularization settings (C, penalty, class_weight) and again tuned the threshold on validation.
- Best tuned threshold ≈ 0.27 (similar to baseline → indicates model calibration remained stable)

Final model: XGBoost (Tree-based)

- I trained a tree-based model using one-hot features, did light hyperparameter tuning on max_depth, min_child_weight, subsample, and colsample_bytree, selected the best model by weighted validation F1, then refit on all 1994 and evaluated once on 1995 (final test).


Part 1: Income Classification Pipeline
Raw CPS features
Replace "?" → NA Fill unknown
One-hot categoricals + (scale nums for LogReg)
Baseline Model Fine tune Base XGBoost
Compare(F1 Score)
XGBoost Final Prediction

(Figure 2: Part 1 Pipeline)

Performance comparison(weighted, val set):

	Baseline model	Baseline Optimal threshold	Tuned log-reg optimal threshold	XGBoost optimal threshold
Accuracy	0.9548	0.9476	0.9468	0.9521
Precision	0.7283	0.5622	0.5550	0.6001
Recall	0.4084	0.6177	0.6268	0.6316
F1 Score	0.5233	0.5886	0.5887	0.6154

The above performance is based on the train test and validation set, which only includes the data from the year 1994.

- **Baseline:** High precision (0.728) but low recall (0.408). When it predicts ≥$50K, it's often right, but it misses many true ≥$50K individuals.
- **Threshold tuning:** Lowering the threshold (~0.28) substantially increases recall (~0.618) at the cost of lower precision (~0.562). This improves weighted F1 from **0.523 → 0.589**, which is more aligned with marketing targeting goals.
- **Tuned Logistic Regression:** Hyperparameter tuning provides only marginal gains over threshold tuning alone (F1 ~0.589 in both cases), suggesting linear model capacity is the limiting factor rather than regularization settings.
- **XGBoost:** XGBoost achieves the best balance of precision and recall on the 1994 weighted validation set (F1 **0.615**), indicating it captures nonlinear interactions among demographic/employment variables that Logistic Regression cannot.

Final Model(XGBoost + Optimal Threshold):

Threshold	0.3
Accuracy	0.9485
Precision	0.6168
Recall	0.6268

F1	0.6218
Roc_auc	0.9509

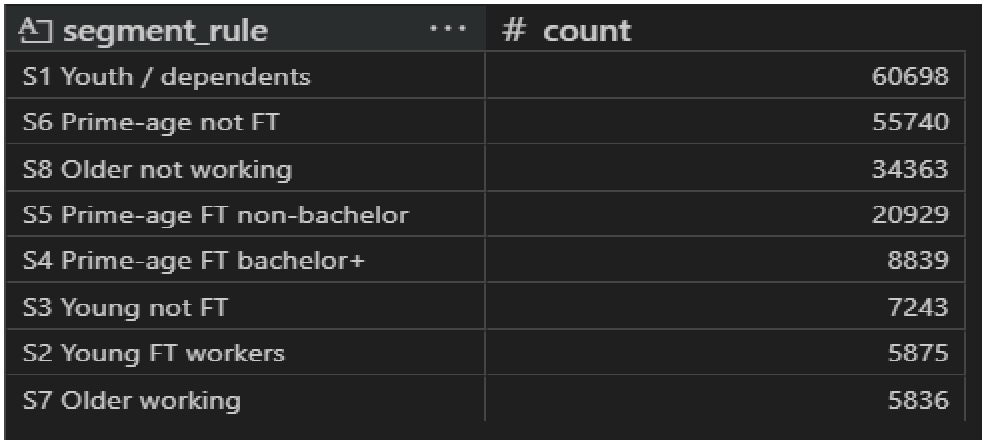
The above final evaluation is trained on all data from the year 1994 and tested on all data from the year 1995.

Part 2: Unsupervised Learning Model for Segmentation

Rule-based segmentation:

I built business-friendly segments using engineered buckets:

- age_bucket (young / working-age / older)
- edu_bucket (child / <HS / HS+some college / bachelor+)
- work_bucket (child/armed forces / full-time/part-time/unemployed / not in labor force)
- I manually segment into 8 different clusters.



segment_rule	# count
S1 Youth / dependents	60698
S6 Prime-age not FT	55740
S8 Older not working	34363
S5 Prime-age FT non-bachelor	20929
S4 Prime-age FT bachelor+	8839
S3 Young not FT	7243
S2 Young FT workers	5875
S7 Older working	5836

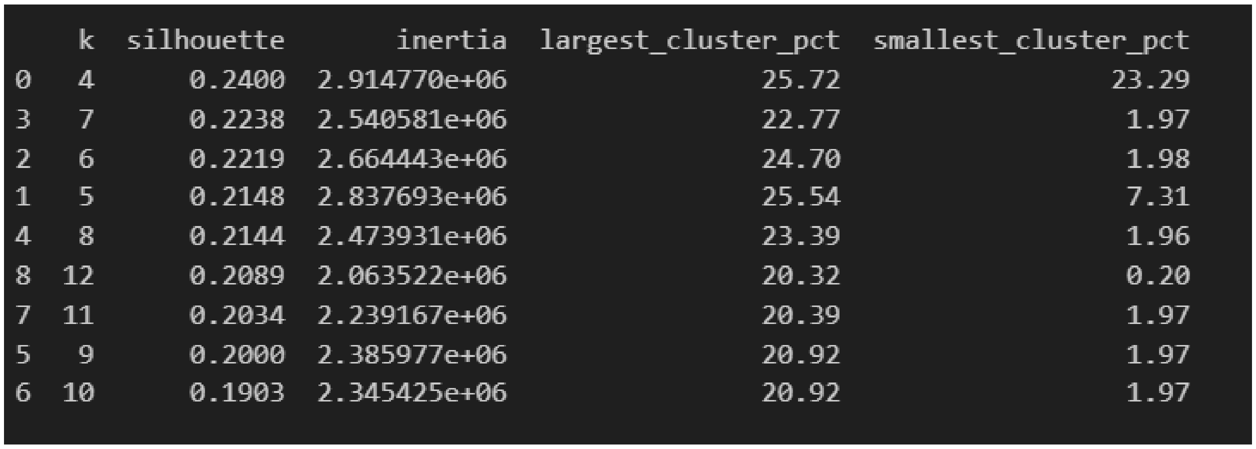
(Figure 3: Rule-based Cluster)

ML Segmentation (K-Means clustering)

- Because clustering in a high-dimensional one-hot space can be noisy and unstable, I use the TruncatedSVD to lower the dimension from 405 to 50.

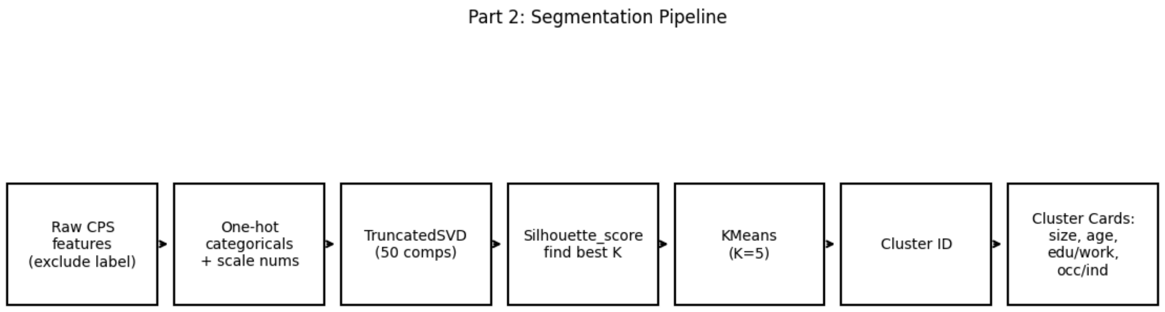
Choosing K (number of clusters)

I tested K=4..12 using silhouette on a random sample and cluster size sanity checks. So we can make sure to balance the separation quality, interpretability, and avoidance of tiny micro-segments.



```
    k  silhouette       inertia  largest_cluster_pct  smallest_cluster_pct
0   4      0.2400  2.914770e+06                25.72                 23.29
3   7      0.2238  2.540581e+06                22.77                  1.97
2   6      0.2219  2.664443e+06                24.70                  1.98
1   5      0.2148  2.837693e+06                25.54                  7.31
4   8      0.2144  2.473931e+06                23.39                  1.96
8  12      0.2089  2.063522e+06                20.32                  0.20
7  11      0.2034  2.239167e+06                20.39                  1.97
5   9      0.2000  2.385977e+06                20.92                  1.97
6  10      0.1903  2.345425e+06                20.92                  1.97
```

(Figure 4: Best K selection result)

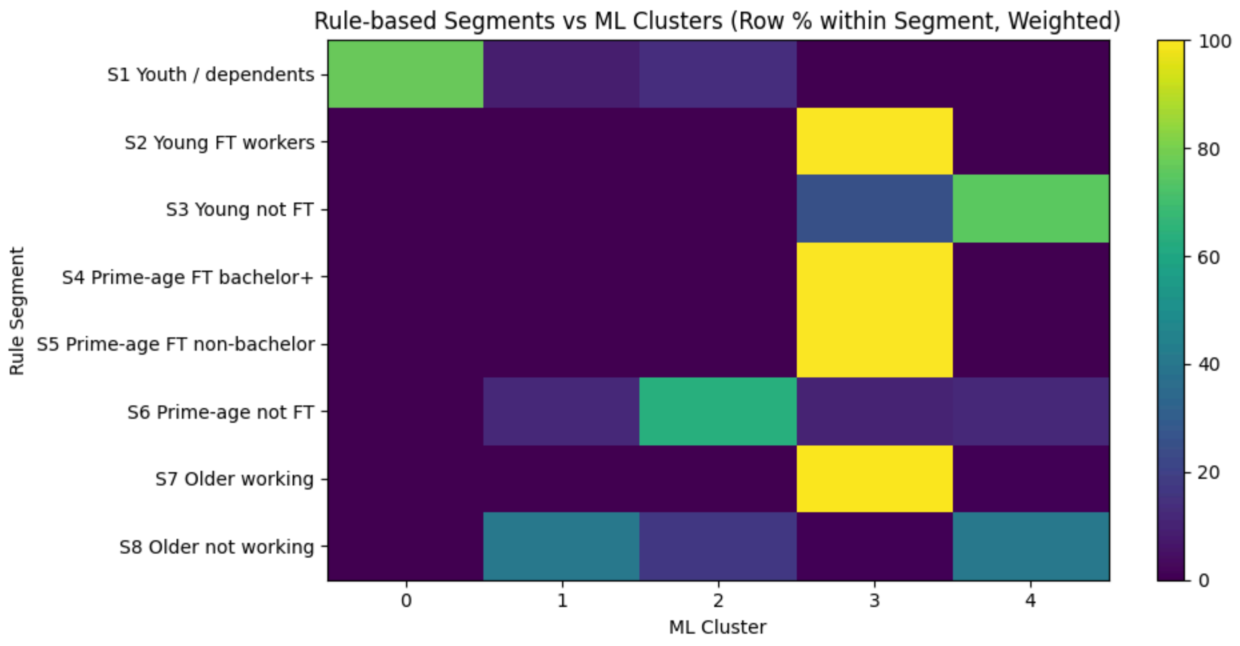

Part 2: Segmentation Pipeline
Raw CPS features (exclude label)
One-hot categoricals + scale nums
TruncatedSVD (50 comps)
Silhouette_score find best K
KMeans (K=5)
Cluster ID
Cluster Cards: size, age, edu/work, occ/ind

（Figure 5: Part 2 Pipeline）

Cluster headline summary (K=5)

- Cluster 0: 7.31% pop, mean age 21.0, ≥$50K rate 0.09%
    - Young Adults / Students & Early Career (low income)
- Cluster 1: 25.54% pop, mean age 38.4, ≥$50K rate 12.44%
    - Prime-Age Workforce (Segment A)
- Cluster 2: 25.34% pop, mean age 38.2, ≥$50K rate 11.14%
    - Prime-Age Workforce (Segment B)
- Cluster 3: 22.77% pop, mean age 6.9, ≥$50K rate 0.00%
    - Children / Dependents
- Cluster 4: 19.05% pop, mean age 62.4, ≥$50K rate 2.09%
    - Older / Retired or Not Working

Heatmap: Rule-based vs ML



（Figure 6: Heatmap）

This heatmap shows that the ML separates different clusters pretty well and largely recovers the structure of the rule-based segmentation.

Recommendation & Future Improvement

Recommendation for Income classifier

Use the XGBoost model as the final model. For a balanced campaign objective, I recommend starting with the **F1-optimal threshold (~0.30)** because it provides a practical tradeoff between wasted spend (false positives) and missed high-income prospects (false negatives). From there, adjust the threshold based on marketing economics: raise it for premium/limited-budget campaigns to improve precision, and lower it for broader reach campaigns to improve recall.

Recommendation for Part 2 (Segmentation model)

Use the segmentation results to drive **message and offer personalization**. The rule-based segments provide an immediately deployable framework (age × work status × education) for simple targeting rules, while the ML clusters (K=5) offer a data-driven view that highlights how people naturally group by life stage and labor-force attachment, and reveals meaningful subtypes within broad categories. I recommend assigning each customer to a segment/cluster at scoring time and using it to choose campaign strategy: premium bundles and loyalty programs for “prime-age workforce” clusters, value-driven offers for young/not-full-time clusters, and convenience/household essentials for older/not-working clusters. All in all, the segmentation layer improves marketing efficiency by tailoring creative, channel, and product selection to group characteristics, and it should be reviewed periodically to confirm clusters remain stable as the customer mix shifts.

Future Improvement

I recommend validating the model’s real business impact through a controlled experiment. For example, run an A/B test where the treatment group receives targeting driven by the income score, while the control group follows the current targeting strategy; measure incremental outcomes such as conversion rate, revenue per impression, and ROI, with guardrails for customer experience (unsubscribe/complaint rate). To make results robust, stratify randomization by key segments (e.g., age/work buckets or ML clusters) and use the model score to define multiple “score bands” (top, mid, low) so you can learn the best operating threshold under real budget constraints. Over time, monitor calibration and segment drift and refresh the model periodically using newer data to maintain performance.