

Income Classification & Customer Segmentation for Retail Marketing

Table of contents

1. Executive Summary
2. Business Problem and Data
3. Dataset Overview (Weighted CPS)
4. Exploratory Data Analysis (EDA)
5. Data cleaning and missing values
6. Feature engineering (core design choices)
7. Encoding and scaling
8. Findings
9. Recursive Feature Elimination:
10. Classification Modeling (Architecture, Training, Evaluation)
11. Segmentation Modeling (K-Means Personas)
12. Business Recommendations (Activation)
13. Risks, Next Steps
14. References

Executive summary

This project explains two analytics solutions:

- (1) a binary income classifier predicting whether an individual's income is $\leq \$50K$ or $> \$50K$,
- (2) a segmentation model that clusters people into interpretable personas for targeted retail marketing.

Because the dataset is derived from a complex survey and includes a person-weight column, we treat weights as a core requirement for population-representative statistics and evaluation

Across benchmark runs, gradient boosting models dominated performance on the classification, with LightGBM (LGBMClassifier) consistently leading by ROC AUC (~0.949) and achieving strong accuracy (~0.876) and F1 (~0.879), making it the recommended scoring model for classification Task and F1-Score metric is prioritized due to the high imbalance in the classes.

Business problem & data

The client's marketing objectives are:

- **Income targeting (classification):** Predict whether a prospect is likely to be a high earner ($> \$50K$) to prioritize premium offers, reduce wasted spend, and support probability-threshold tuning aligned to campaign ROI.
- **Persona development (segmentation):** Identify customer groups that differ in demographics and economic signals to tailor messaging, channels, and offers beyond a single income label.

Dataset and why it is weighted

The dataset is census/CPS-derived “Census-Income (KDD)” style data containing ~40+ demographic and employment attributes (e.g., age, education, occupation/industry codes, marital status) plus a binary income. A weight column indicates how many people in the population each record represents, which is a standard feature of survey microdata designed to make sample estimates representative of the full population. Using weights is essential because ignoring them can bias descriptive conclusions and distort performance estimates for the population the campaign is targeting.

Exploratory data analysis (EDA)

EDA focused on understanding data structure, “not in universe” categories, skew/zero-inflation in wage and investment features, and high-cardinality code fields that may add noise without interpretability.

Key EDA themes:

- **High cardinality fields:** Detailed occupation/industry codes can have very high cardinality; EDA flagged them as candidates for removal or constrained use to avoid sparse, noisy modeling.
- **Skewed numeric distributions:** Wage and investment-related fields typically show heavy right tails and many zeros, motivating robust models and engineered aggregation features.

Steps in Detail:

1. Renamed Column names for better formatting
2. Removed Columns which have “Not in Universe” as value for more than 51% of the data records:

```

Count of 'Not in universe' records per column:
vet_question      99.005628
unemp_reason     96.957744
hs_college        93.694962
state_prev_res    92.094646
region_prev_res   92.094646
union_member      90.445212
fam_under_18      72.288408
mig_same          50.726984
major_ind_code    50.462353
major_occ_code    50.462353
class_worker       50.242328
mig_prev_sunbelt  42.127474
mig_move_reg      0.759812
mig_chg_msa       0.759812
mig_chg_reg       0.759812
dtype: float64

```

Columns removed:

```

['vet_question' , 'unemp_reason' , 'hs_college' ,
'state_prev_res' , 'region_prev_res' , 'union_member' ,
'fam_under_18'

```

3. Dropped Non Informative columns :

- Dropped “det_hh_fam_stat” as “det_hh_summ” was more informative
- Dropped “mig_chg_msa”, “mig_chg_reg”, “mig_move_reg”, “mig_same”, “mig_prev_sunbelt” as data was not informative
- Dropped “country_father”, “country_mother”, “country_self”, “citizenship” as I felt it had no impact on earning
- Dropped codes : “det_ind_code”, “det_occ_code”

Data cleaning and missing values

The notebook applies practical cleaning steps commonly used for census-style tabular datasets:

- Replacing “?” placeholders with missing values (NaN).
 - Inspecting missingness by column.
 - Dropping columns with missingness that were not handled via imputation in the baseline pipeline (with an explicit note that categorical imputation via “Unknown” is a viable future enhancement).
-

Feature engineering (core design choices)

This project uses feature engineering to convert raw survey fields into model-friendly signals and reduce redundancy.

(a) Education → ordinal numeric (edu_year)

Education is ordinal (e.g., high school < bachelor’s < master’s), so the pipeline maps education categories into a numeric “years” scale (edu_year) to preserve ordering and support both classification and clustering.

```
'Children': 0,
'Less than 1st grade': 0.5,
'1st 2nd 3rd or 4th grade': 2.5,
'5th or 6th grade': 5.5,
'7th and 8th grade': 7.5,
'9th grade': 9,
'10th grade': 10,
'11th grade': 11,
'12th grade no diploma': 12,
'High school graduate': 12,
'Some college but no degree': 14,
'Associates degree-academic program': 14,
'Associates degree-occup /vocational': 14,
'Bachelors degree(BA AB BS)': 16,
'Masters degree(MA MS MEng MEd MSW MBA)': 18,
'Prof school degree (MD DDS DVM LLB JD)': 20,
'Doctorate degree(PhD EdD)': 21
```

(b) Composite income proxy (total_income)

```
# Full-time > 1.0
'Full-time schedules': 1.0,
# Part-time -> 0.5
'PT for non-econ reasons usually FT': 0.5,
'PT for econ reasons usually PT': 0.5,
'PT for econ reasons usually FT': 0.5,
# Unemployed / Not in Labor Force / Other -> 0.0
'Unemployed full-time': 0.0,
'Unemployed part-time': 0.0,
'Not in labor force': 0.0,
'Children or Armed Forces': 0.0
```

A composite total_income feature is engineered to consolidate multiple sparse and skewed income-related signals (wage/work intensity plus investment-like components) into a single continuous variable intended to better reflect economic status than any single raw field.

```
'total_income' = ('wage_per_hour' * 'full_or_part_emp' * 'weeks_worked' * 80) +
'capital_gains' - 'capital_losses' + 'stock_dividends'
```

(c) Redundancy reduction

After building total_income, the original component columns (wage/work fields and investment subcomponents) are removed to reduce multicollinearity and simplify the feature space for both modeling and segmentation.

Encoding and scaling

Categorical encoding

Categorical variables are label-encoded after basic string cleanup (e.g., trimming whitespace), which is a pragmatic choice for tree-based boosting models that can often work effectively with integer-encoded categories in practice.

Scaling

Numeric variables are standardized (e.g., StandardScaler) prior to distance-based methods (K-Means) and to avoid magnitude dominance of variables like income proxies compared to age.

Weighted statistics and why they matter

The workflow computes weighted descriptive statistics to reflect population-representative distributions rather than sample-only statistics.

Statsmodels' DescrStatsW supports descriptive statistics for case weights, aligning with the project's need to treat weights as representing differential population counts per record. This ensures summary statements (means/quantiles/top categories) are aligned with the survey design and the weight semantics.

Understanding from the Dataset:

1. Population Overview

Scale: The data represents a weighted population of approximately 347.2 million individuals.

Age Structure: The population is relatively young, with an average age of 34.5 years and a median of 33 years. The range spans from infants (0) to seniors (90).

2. Economic Profile

Extreme Income Skew: There is a massive disparity in Total Income. While the weighted average (mean) is roughly \$101,368, the median (50th percentile) is \$0.

Interpretation: This suggests the dataset includes a large number of non-earners (likely children or non-working spouses) who pull the median down, while a small number of very high earners (max ~\$40.7M) pull the average up significantly.

3. Demographics

Nativity: The vast majority are "Native- Born in the United States" with "United-States" being the most common origin for parents as well.

Composition: The most frequent race category is White, and the most frequent sex is Female.

4. Data Coverage ("Not in Universe")

Missingness/Applicability: For many complex variables (Class of Worker, Tax Filer Status, Union Member, Migration), the top category is "Not in universe."

Implication: This indicates that a significant portion of the population (likely children or those outside the labor force) was not eligible for these specific questions.

Findings

1) Weight handling and imbalance strategy (critical design)

This dataset is both weighted and imbalanced in the target label (> \$50K is the minority class), so the workflow addresses both issues together.

Implementation-level decisions:

- **Train/test split with stratification** to preserve label proportions, while carrying weights through the split so evaluation remains representative.
- **Weight-consistent imbalance correction** by rebalancing class weight totals in training so the model does not learn a trivial “always majority class” solution.
Performed down Sampling of Majority class weights to match the minority class

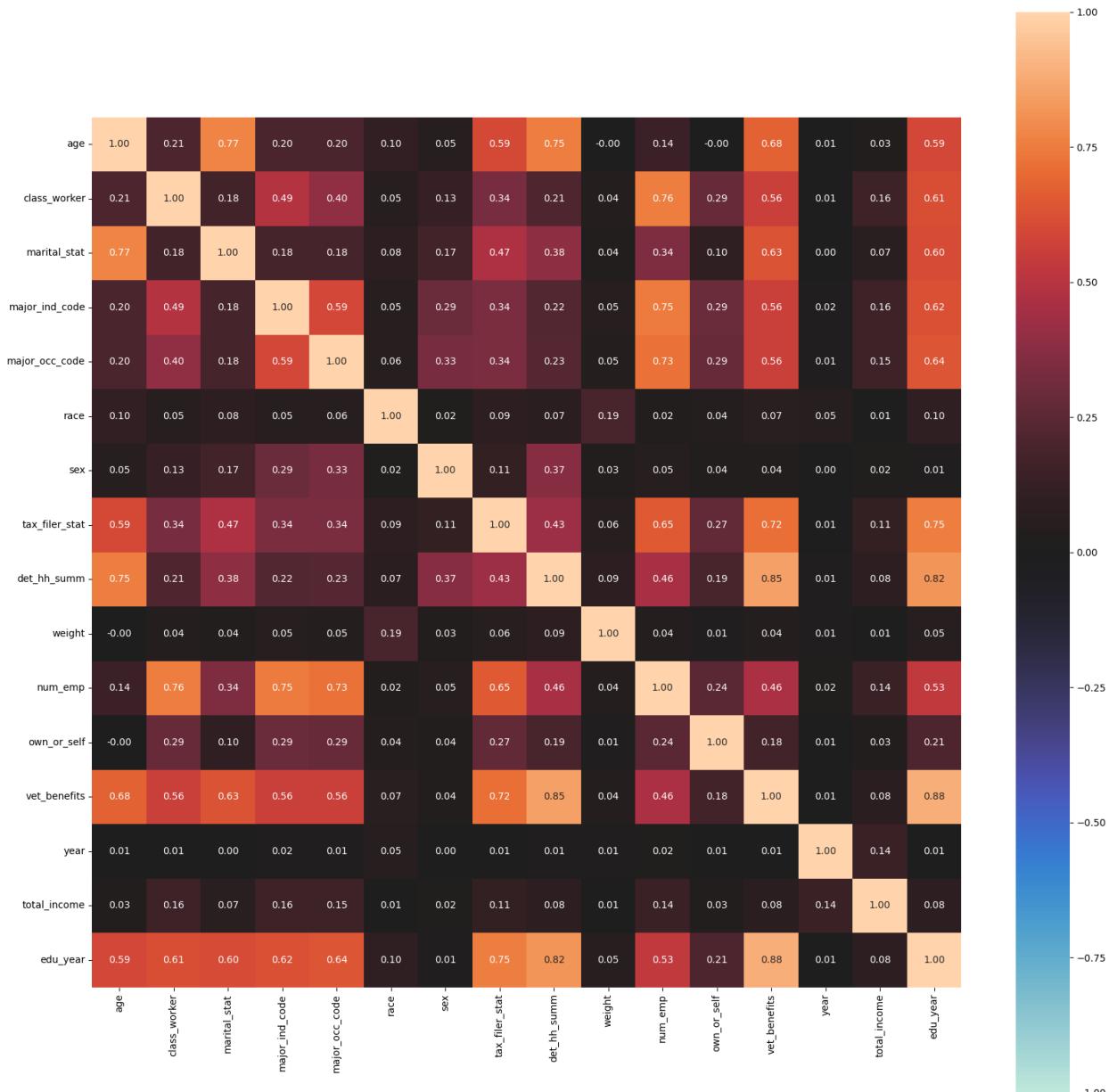
```
Total Weight Class 0: 260,051,546
Total Weight Class 1: 17,757,765
Imbalance Ratio: 0.07
New Weighted Sums (Should be approx equal):
income_50k
0    17757764.69
1    17757764.69
Name: weight, dtype: float64
income_50k
0    4.435359e+06
1    4.483481e+06
Name: weight, dtype: float64
```

- **Models without native weight support** handled through weighted-probability resampling of the training data, while retaining weight-aware evaluation on the original test set.

Why this matters for marketing:

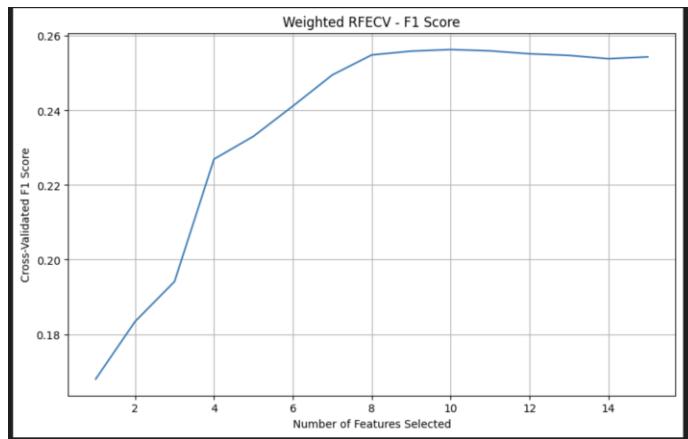
- The high-income class is usually the higher-value marketing segment, so recall/precision tradeoffs must be controlled intentionally rather than dictated by class prevalence.

2) correlation Analysis: found No correlation



Recursive Feature Elimination:

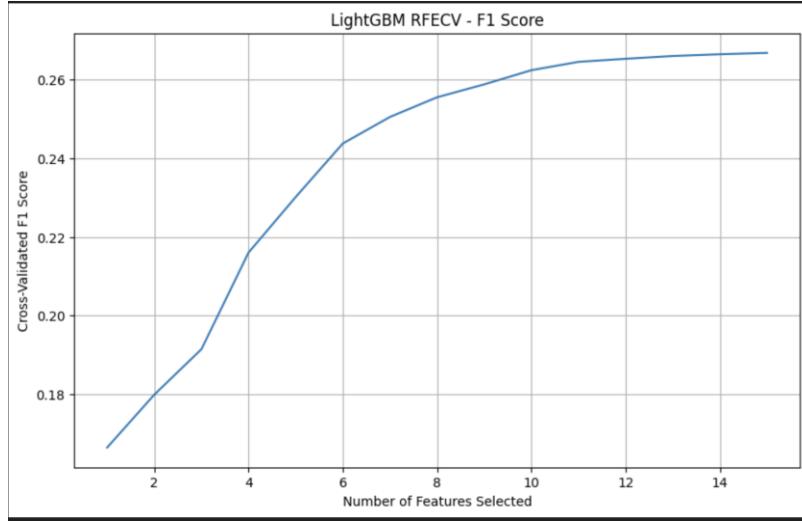
1. Random Forest model as feature selector with 5 fold cross validation to identify feature importance



Found Important columns to be :

```
[ 'age',
  'class_worker',
  'major_ind_code',
  'major_occ_code',
  'tax_filer_stat',
  'sex',
  'num_emp',
  'det_hh_summ',
  'edu_year',
  'total_income']
```

2. LightGBM model as feature selector with 5 fold cross validation to identify feature importance:



Feature Importance:

Top 10 Selected Features by Importance:		
	Feature	Importance
0	age	1.538764e+08
1	edu_year	5.683106e+07
2	total_income	4.361294e+07
3	num_emp	2.883007e+07
4	sex	2.055807e+07
5	major_occ_code	1.815408e+07
6	tax_filer_stat	1.167938e+07
7	det_hh_summ	1.163953e+07
8	class_worker	9.863290e+06
9	major_ind_code	9.487445e+06
10	marital_stat	3.784651e+06
11	race	3.068313e+06
12	year	2.407780e+06
13	own_or_self	2.375520e+06
14	vet_benefits	3.641645e+05

Classification modeling

Models evaluated

The notebook benchmarks a broad suite of classifiers to compare linear baselines, tree-based models, ensembles, and gradient boosting methods, including:

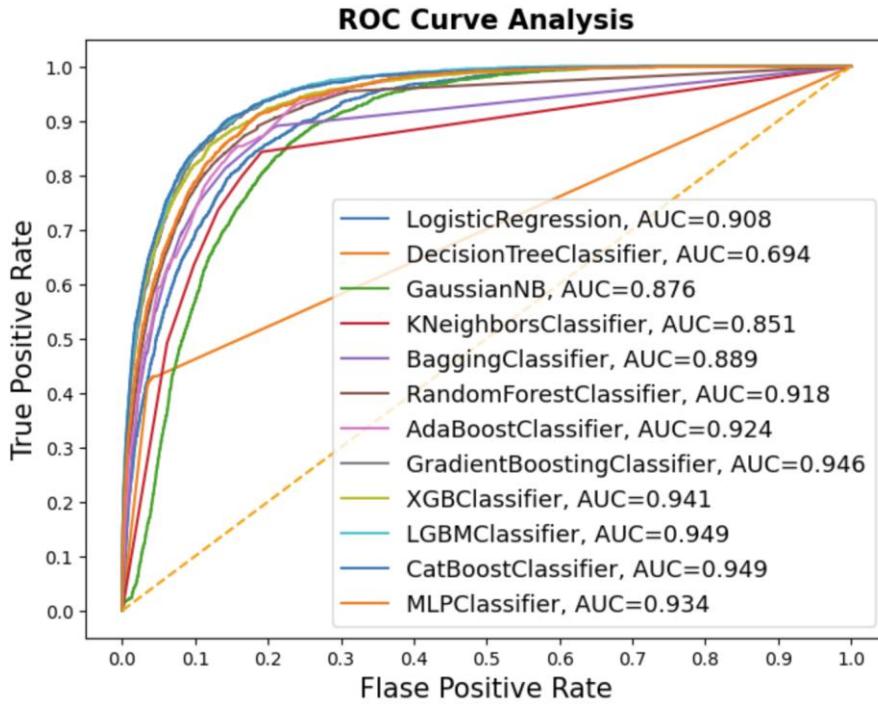
- Logistic Regression, Decision Tree, Random Forest
- AdaBoost, Gradient Boosting
- XGBoost, LightGBM, CatBoost
- Gaussian Naive Bayes, KNN, MLP (with special handling when weights are not supported)

Metrics and evaluation

F1 Score is prioritized because it considers both precision and recall and due to huge class imbalance it's a better metric rather than accuracy. ROC AUC is emphasized because it measures ranking/discrimination and supports downstream threshold selection for campaign operations. Additional metrics include accuracy, precision/recall, log loss, and confusion matrices (including weighted reporting where applicable).

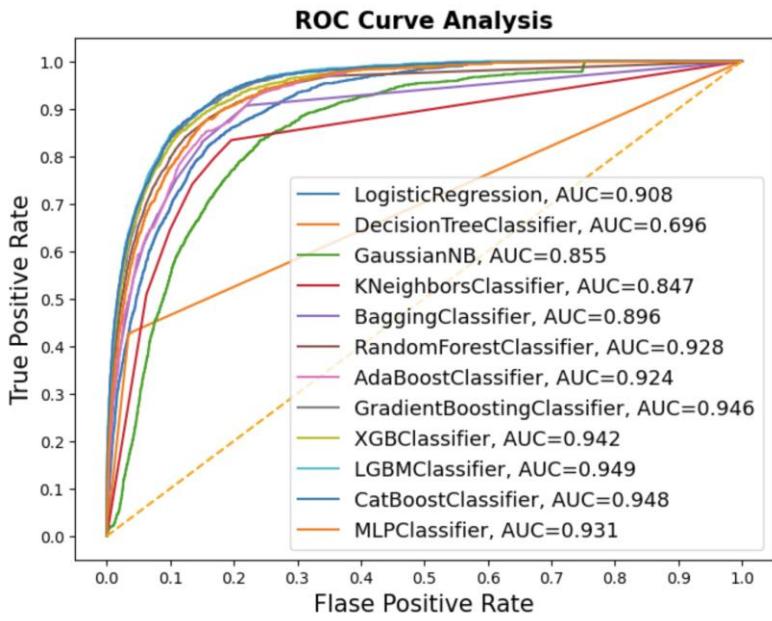
1. Test Data evaluation results using Random Forest features selected

classifiers	Roc Auc	Accuracy	f1 Score	logloss	Confusion Matrix	Precision	Recall
LogisticRegression	0.907677	0.828930	0.835085	0.378599	[[3530125.135141388, 905233.6054651847], [6205...	0.810152	0.861601
DecisionTreeClassifier	0.694155	0.692393	0.584580	10.933717	[[4245020.718130956, 190338.02247562417], [255...	0.910246	0.430541
GaussianNB	0.876206	0.814830	0.828061	0.618030	[[3290513.0958625367, 1144845.644744034], [506...	0.776470	0.886996
KNeighborsClassifier	0.851227	0.802061	0.789036	4.107876	[[3852055.880024839, 583302.860581738], [11820...	0.849846	0.736347
BaggingClassifier	0.889406	0.752141	0.691878	2.423292	[[4226289.117782247, 209069.62282434394], [200...	0.922308	0.553573
RandomForestClassifier	0.918275	0.686252	0.557614	1.444101	[[4357014.091911875, 78344.64869471811], [2719...	0.957465	0.393346
AdaBoostClassifier	0.924377	0.844492	0.846592	0.514118	[[3704906.9736152883, 730451.7669912841], [656...	0.839723	0.853574
GradientBoostingClassifier	0.946145	0.873389	0.877104	0.295664	[[3760035.251906197, 675323.4887003679], [4538...	0.856464	0.898762
XGBClassifier	0.940952	0.867397	0.866528	0.324507	[[3897125.920398494, 538232.8202080758], [6444...	0.877039	0.856265
LGBMClassifier	0.949279	0.876133	0.878653	0.285411	[[3814415.6162432604, 620943.1243633066], [483...	0.865615	0.892091
CatBoostClassifier	0.948597	0.874732	0.874347	0.293714	[[3914437.0077119675, 520921.7328945986], [596...	0.881826	0.866995
MLPClassifier	0.934121	0.861022	0.868747	0.334251	[[3577180.213445988, 858178.5271605767], [3813...	0.826991	0.914944



2. Test Data evaluation results using XGBoost features selected:

classifiers	Roc Auc	Accuracy	f1 Score	logloss	Confusion Matrix	Precision	Recall
LogisticRegression	0.908402	0.830705	0.836706	0.377242	[[3540559.4782992066, 894799.262307372], [6151...	0.812142	0.862803
DecisionTreeClassifier	0.696077	0.694046	0.584991	10.961576	[[4266881.125153903, 168477.61545269113], [256...	0.919454	0.428954
GaussianNB	0.855078	0.766970	0.798522	1.160587	[[2721902.1030364214, 1713456.6375701516], [36...	0.706199	0.918613
KNeighborsClassifier	0.847398	0.803030	0.790972	4.275505	[[3838302.801426848, 597055.939179726], [11596...	0.847723	0.741343
BaggingClassifier	0.896244	0.749884	0.686368	2.123522	[[4247160.46177175, 188198.27883483336], [2042...	0.928418	0.544429
RandomForestClassifier	0.928305	0.676933	0.534607	1.193723	[[4382503.573112535, 52855.16749405353], [2828...	0.969051	0.369122
AdaBoostClassifier	0.924377	0.844492	0.846592	0.514118	[[3704906.9736152883, 730451.7669912841], [656...	0.839723	0.853574
GradientBoostingClassifier	0.946192	0.875162	0.879127	0.295756	[[3756419.8403027514, 678938.9003038126], [434...	0.856399	0.903095
XGBClassifier	0.942456	0.867678	0.866476	0.319531	[[3909502.9152761805, 525855.8253303878], [654...	0.879253	0.854065
LGBMClassifier	0.949277	0.873774	0.875818	0.285449	[[3823134.749118014, 612223.9914885563], [5135...	0.866389	0.885454
CatBoostClassifier	0.948008	0.872805	0.871522	0.297217	[[3936707.7330881855, 498651.00751838036], [63...	0.885271	0.858194
MLPClassifier	0.931345	0.859024	0.861829	0.348037	[[3740248.0066915057, 695110.7339150651], [562...	0.849425	0.874600



Results (from benchmark tables shown in the reports/notebook outputs)

Top results were consistently achieved by gradient boosting variants, with LightGBM leading by ROC AUC.

Representative reported metrics:

- **LGBMClassifier:** ROC AUC ≈ 0.949 ; Accuracy ≈ 0.876 ; F1 ≈ 0.879
- **CatBoostClassifier / GradientBoosting / XGBoost:** very close runners-up by ROC AUC

Why boosting models dominate here:

- Census/career/income relationships are nonlinear with complex interactions (education \times occupation \times age \times household signals), and gradient-boosted decision trees are well known to perform strongly on structured/tabular data.

Final classifier recommendation

Recommendation: Deploy LightGBM (LGBMClassifier) as the production scoring model for lead ranking and probability-based targeting. LightGBM is a gradient boosting decision tree system designed for efficiency and performance, making it a strong fit for tabular classification with mixed feature types. Operationally, marketing should set the probability threshold based on campaign economics (e.g., contact cost, conversion lift, and acceptable false-positive rate), rather than optimizing only for accuracy at a fixed 0.5 threshold

Segmentation modeling (K-Means personas)

K-Means

K-Means is a standard, interpretable clustering method that partitions observations into k clusters by minimizing within-cluster distances, and it is widely used to create actionable personas from scaled numeric features.

It supports a clear workflow: choose k, fit clusters, interpret centroids, and translate centroid differences into marketing actions.

Features used for segmentation

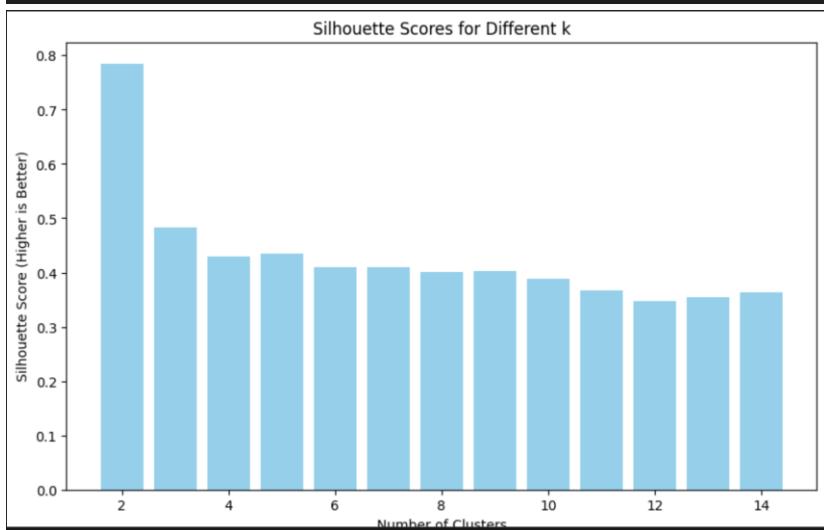
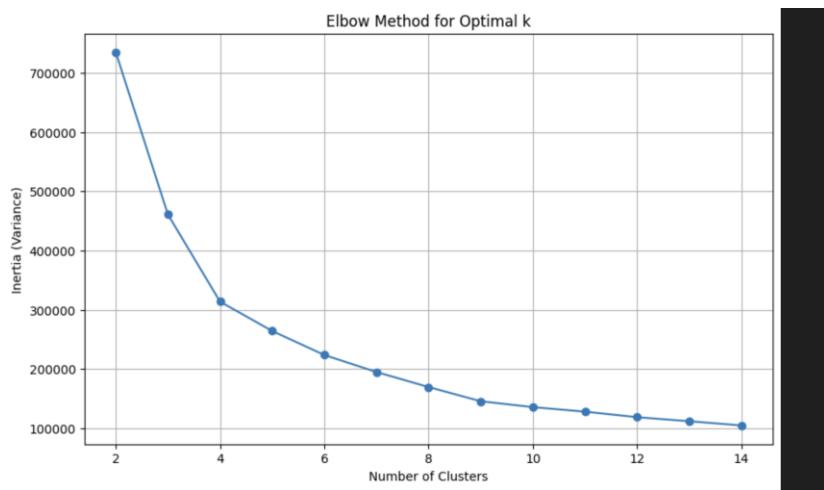
The segmentation dataset focuses on a marketing-relevant subset including:

- Age (life stage proxy)
- Education years (edu_year) (human capital proxy)
- Employment-related signal(s) (e.g., num_emp)
- Engineered total_income (economic capacity proxy)
- (In reporting) the observed income label is used to compute “% high earners by cluster” for interpretability (not as an input driver for clustering in a strict unsupervised sense, depending on implementation choice).

All numeric inputs are standardized prior to K-Means to ensure no single high-magnitude feature dominates Euclidean distance.

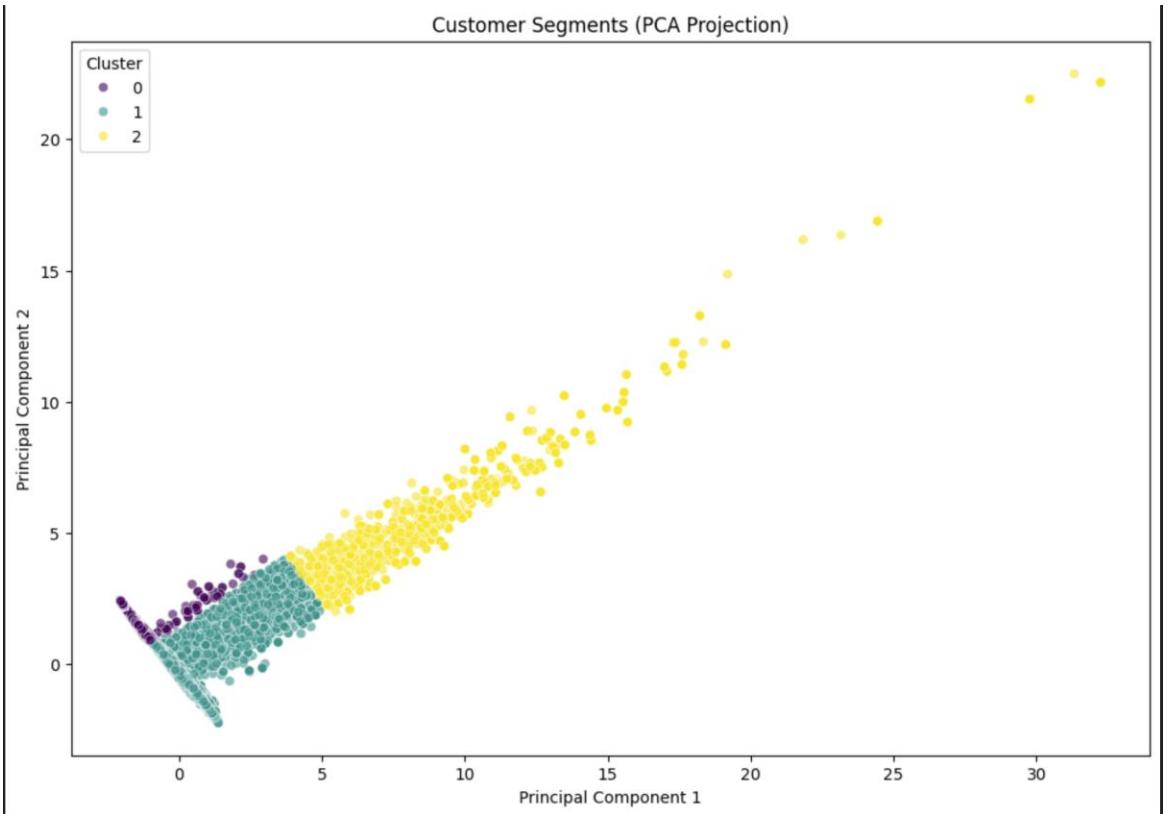
Selecting k and validating separability

The notebook uses cluster-quality heuristics (including elbow and silhouette-style reasoning) and proceeds with **k = 3** as the final segmentation choice, balancing interpretability and separation.



Silhouette analysis is a standard approach for assessing cluster cohesion/separation and guiding selection of n_clusters in K-Means.[scikit-learn](#)

A PCA-to-2D visualization is used to visually sanity-check whether clusters are reasonably separable in a compressed space.



Cluster profiles and marketing personas (interpretable activation)

Based on the cluster centroid summaries and the percentage of >\$50K earners by cluster reported in the notebook outputs, the clusters map naturally to the following personas:

--- Marketing Segmentation Profiles ---			
Cluster	0	1	2
age	-1.03	0.50	0.37
num_emp	-0.81	0.62	1.20
edu_year	-1.89	0.36	0.30
total_income	-0.14	-0.08	10.82
income_50k	0.00	0.60	0.68
Count	33532.00	171898.00	2720.00
Percent	16.11	82.58	1.31

- Persona A: **Value-seeking / lower-income**

- Generally younger and/or lower edu_year, lower total_income centroid, lowest share of >\$50K.
 - Activation: discount-led offers, value packs, price-sensitive messaging, channels with lower acquisition cost.
- Persona B: Mainstream / stable
 - Largest cluster with moderate centroids and a meaningful share of >\$50K.
 - Activation: loyalty programs, bundles, cross-sell/upsell, standard product assortment and seasonal campaigns.
- Persona C: High-income / high value
 - Smallest cluster with the highest total_income centroid and highest share of >\$50K.
 - Activation: premium product lines, concierge-style experiences, exclusive promotions, higher-touch channels, and margin-optimized offers.

Risks, Compliance and next steps

How marketing should use the outputs (operating model)

Recommended operational pattern:

1. **Score new leads** with the LightGBM classifier to produce $P(\text{income} > 50K)$.
2. **Choose a threshold** aligned to budget and ROI (e.g., only contact leads with probability ≥ 0.8 for premium campaigns; use a lower threshold for broader awareness campaigns).
3. **Overlay segmentation personas** to tailor creative, product, and channel strategies (premium vs value, high-touch vs low-touch).
4. **Measure lift and recalibrate** thresholds and messaging based on observed conversion and revenue outcomes.

Risks and limitations

- **Temporal drift:** The dataset reflects CPS-era feature relationships (1994–1995), so performance may degrade on modern populations unless recalibrated or retrained with contemporary data.
 - **Survey-weight sensitivity:** Because each row can represent many people, errors may disproportionately affect population-level conclusions if weights are ignored or mishandled.
 - **Fairness and compliance:** Income prediction and marketing activation can correlate with sensitive characteristics; usage should be reviewed for legal, regulatory, and reputational risk, with clear governance around permissible targeting attributes and messaging.
-

Recommended enhancements

Modeling improvements:

- Hyperparameter optimization for LightGBM/CatBoost (e.g., Optuna or GridSearch) to capture incremental uplift.
- Explainability package (feature importance + SHAP-style local/global explanations) to support stakeholder trust, QA, and compliance documentation.
- Probability calibration (Platt scaling / isotonic regression) to improve reliability of predicted probabilities for threshold-based decisions.

Segmentation improvements:

- Segment stability checks (bootstrap re-fitting, centroid variance, silhouette monitoring) to ensure personas are robust enough for long-lived campaigns.
- Consider alternative clustering (Gaussian Mixture Models or hierarchical clustering) if personas need non-spherical clusters or probabilistic assignment.

Data pipeline improvements:

- Replace “drop missing column” with categorical imputation (“Unknown”) for fields like hisp_origin to preserve signal while keeping the pipeline deterministic.
- Add monitoring for feature drift and score drift if deploying into a live lead-scoring system

References

- UCI Machine Learning Repository, “Census-Income (KDD)” dataset description.
- U.S. Census Bureau, CPS weighting methodology overview (weights relate to inverse selection probability and population representation).
- Statsmodels documentation for weighted descriptive statistics (DescrStatsW).
- LightGBM paper: “LightGBM: A Highly Efficient Gradient Boosting Decision Tree.”
- Scikit-learn example: selecting number of clusters with silhouette analysis for K-Means.
- K-Means overview and algorithm intuition (centroid-based clustering).