2025

Census Analysis for Marketing

CLASSIFICATION AND SEGMENTATION

VIKRAM PANDE

Introduction

The report details a machine learning project aimed at developing predictive and segmentation models to enhance a retail client's marketing strategy. The primary objective is to leverage a rich dataset, containing 40 demographic and employment-related variables, to effectively target two distinct customer groups based on their income: those earning less than $50,000 and those earning more than $50,000.

EDA

The initial phase of this project focused on **Exploratory Data Analysis** to understand the dataset, its characteristics, and its suitability for the subsequent modeling tasks. The dataset comprises **40 variables**, and the EDA process guided the feature engineering and preprocessing decisions.

1. Data Cleaning and Preparation

- **Feature Naming:** Feature names were standardized for consistency and ease of use throughout the modeling process.
- **Missing Values and Uniqueness:** A comprehensive check for null values and value counts was performed. Only the **hispanic_origin** feature was found to contain missing values, which were imputed with the category **"Do not know"** to retain these records.
- **Handling "Not in Universe" Values:** Several features contained a **"Not in universe"** category (e.g., in unemployment reason). We made the critical decision **not to drop** these features. This value is assumed to be a strong indicator within census data, as it signifies that the attribute is irrelevant for that specific observation (e.g., *Reason for unemployment* only applies to the unemployed). Keeping these features allows the model to leverage this implicit information.

2. Target and Feature Distribution Analysis

- **Target Imbalance:** As expected with census income data, the target variable was found to be **extremely imbalanced**. This is crucial and needs evaluation metrics and potential techniques like oversampling or class weighting during modeling.
- **Income and Wage Skewness:** Analysis of wage-related features revealed they are highly **skewed**. This is acceptable, as the dataset encompasses a diverse range of worker classes and income levels.
- **Correlation Analysis:** The correlation map between the numerical features was analyzed to identify any collinear variables. No strong correlation was found.

3. Feature Importance and Selection

- **Scaling Requirements:** Some of the features are extremely skewed hence the need of scaling was assessed. For Linear models, scaling is required but for Tree Based models, scaling is handled implicitly.
- **Initial Feature Importance:** To quickly understand feature importance, a **Random Forest Classifier** was trained. The feature importance plot generated from this initial model provided an initial ranking of the most predictive features.

Based on all these exploratory steps, specific **feature engineering** transformations were applied for both the classification and segmentation models. You can find the detailed analysis in *1_EDA.ipynb* file

Feature Engineering

Classification Model Preparation

Feature engineering for the classification task was performed following the EDA. This involved renaming columns and handling missing values, as previously detailed.

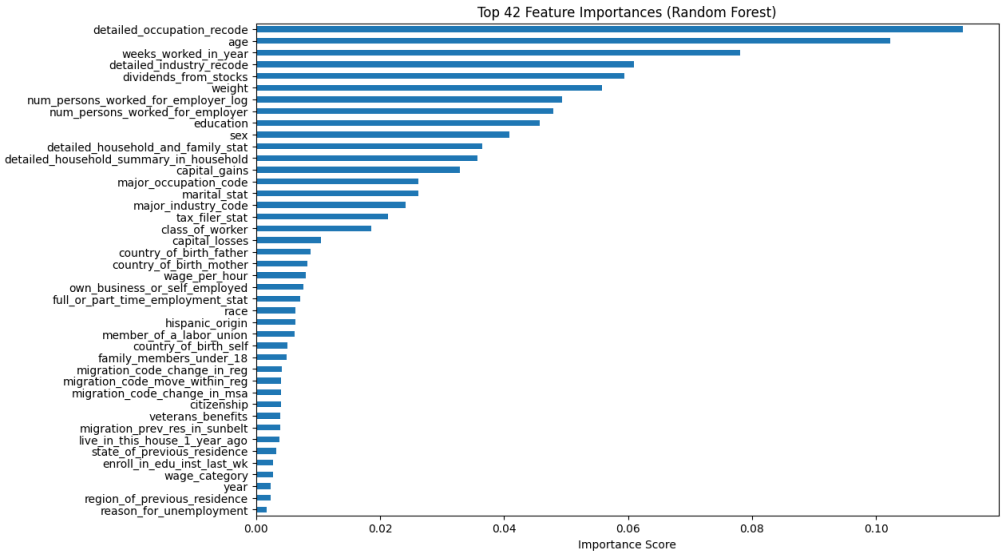
- The target variable was mapped to binary numerical values (0 and 1) instead of the original categories.
- A new, composite feature, **net_gains**, was engineered by combining the values from capital_gains, dividends_from_stocks, and capital_losses.
- Based on the initial feature importance analysis, columns with negligible predictive contribution were dropped to simplify the model and improve performance.
- For **linear models**, numerical features were scaled appropriately. Categorical features were processed using **One-Hot Encoding** and **Ordinal Encoding**, based on the algorithm.

Segmentation Model Preparation

The feature engineering steps for the segmentation model were almost same for those of the classification model, with a few differences:

- For segmentation, a smaller subset of features was used. Specifically, additional features were dropped to retain only the most **quantitative and interpretable columns**.
- Before feeding the features into dimensionality reduction (PCA) and clustering algorithms, all numerical features were scaled using **MinMaxScaler**, and categorical features were transformed using **One-Hot Encoding**.

The detailed reasoning supporting the specific feature engineering choices and column dropping across both models can be found in the preprocessing sections of the notebooks: *2_classification.ipynb* and *3_segmentation.ipynb*.



Feature Importance with Random Forests

Modeling

Classification

Train, Test, and Validation Split

The dataset was initially divided into training, validation, and testing partitions. The training and validation sets were used for offline model development and evaluation, while the final, unseen test set was reserved for inferring the model's performance.

To address the **extreme class imbalance** in the dataset, **SMOTE** was applied. While it is best practice to integrate SMOTE within a model pipeline, computational limitations necessitated applying SMOTE directly to the training data *before* fitting the models. Importantly, SMOTE was **only applied to the training data** to prevent data leakage into the validation or test sets.

Regarding sample weighting (weights feature in the dataset), if SMOTE is utilized, the original weights column becomes unnecessary for prediction, as SMOTE effectively handles the imbalance through synthetic samples. The weights would only have been necessary had SMOTE not been applied.

The data was first split into a train_full set and a test set, with the test set comprising **15%** of the original data. The train_full set was then further partitioned into the final train and validation sets, with the validation set being **17%** of the train_full data. Both the test and validation splits were **stratified** to ensure the income class distribution was preserved across all subsets.

Training with Hyperparameter Search

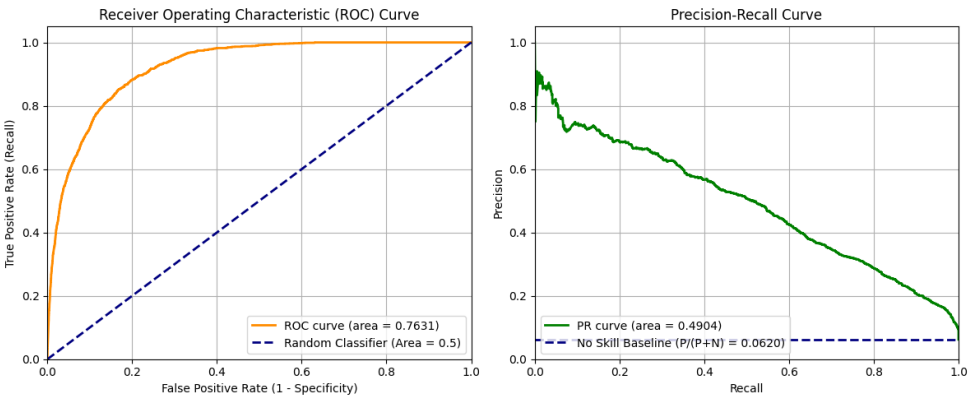
Decided to use and compare five distinct classification models, including linear, tree-based, and ensemble methods: **Logistic Regression, Random Forest Classifier, Histogram Gradient Descent, XGBoost, and LightGBM**.

A comprehensive hyperparameter search space was defined for each model. We then utilized **Stratified Cross-Validation** to test all combinations of parameters. After all models were fitted, the results were sorted based on the primary performance metric, **ROC-AUC**, to select the optimal model. The best-performing model identified was the **Histogram Gradient Descent Classifier**, achieving an ROC-AUC of **$0.9774** on the cross-validation folds.

For final classification, a **dynamic thresholding** technique was also used instead of the default 0.5 threshold to maximize the **F1 score**. The best-performing model was then used for final training and saved for future inference.

Evaluation

Test Metrics: {'Accuracy': 0.9302, 'Precision': 0.4508, 'Recall': 0.5724, 'F1': 0.5044, 'ROC_AUC': 0.7631, 'PR_AUC': 0.2846}



General classification metrics such as **F1 score, accuracy, precision, and recall** were utilized for evaluation. Given the extreme class imbalance, where the positive class constitutes only approximately 6.2 of the samples, we placed a

higher emphasis on the **Precision-Recall (PR) curve** and **PR AUC** as robust performance measures for the minority class.

The model demonstrates reasonable performance on the highly imbalanced dataset, showing strong discriminative power. The test set **ROC-AUC of 0.7631** is a decent result, indicating a significant improvement over random guessing. We acknowledge that the **93% accuracy** is misleading due to the severe imbalance and should not be solely relied upon.

The model achieved a **recall of 57%**, meaning it correctly identified 57% of the true positive cases (high earners). A **precision of 45%** suggests that when the model predicts an instance is a high earner, it is correct about 45% of the time. The resulting **F1 score of 50%** is considered a good balance between precision and recall for this imbalanced problem. The ROC-AUC of 0.76 confirms the probability that the model will rank a randomly chosen positive instance higher than a randomly chosen negative instance.

Segmentation

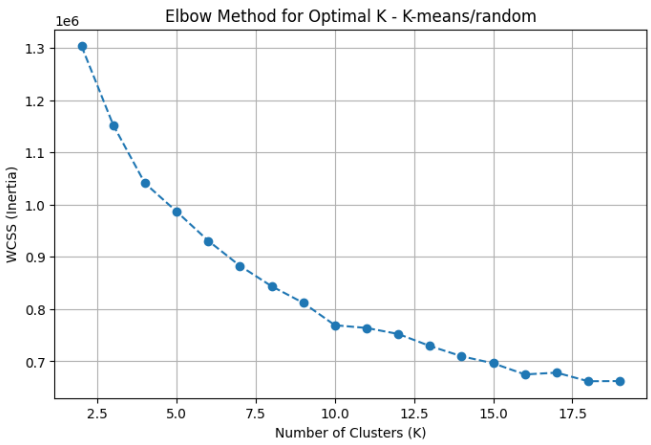
Principal Component Analysis (PCA)

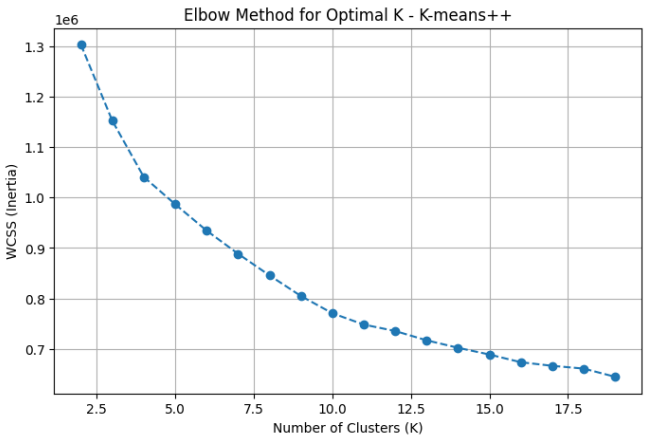
Following the initial feature engineering, the dataset still contained a large number of dimensions, which would negatively impact the efficiency and interpretability of the clustering algorithm. To address this, **Principal Component Analysis (PCA)** was utilized for dimensionality reduction. PCA was selected to reduce the complexity of the feature space while retaining the maximum amount of variance. Set the explained variance threshold at **90%**. This process successfully reduced the data dimensions from the original 1**69 columns** to **43 principal components**, significantly simplifying the input for the subsequent clustering step:

While other techniques like t-SNE and UMAP are also viable for dimensionality reduction, PCA was chosen here for its efficiency and ability to maintain global data structure.

Determining Optimal Clusters (K) using the Elbow Method

To determine the optimal number of clusters K for the K-Means algorithm, the **Elbow Method** was implemented. This method uses the **Within-Cluster Sum of Squares (WCSS)**, or inertia. The "elbow" point is identified where adding an additional cluster no longer provides a significant reduction in WCSS, indicating that the maximum benefit of adding more clusters has been achieved. Tested a range of K values from 2 to 20 using both the standard random initialization and the more robust **k-means++** initialization. Based on the analysis of the WCSS plot, the optimal number of clusters was determined to be **K = 10**.





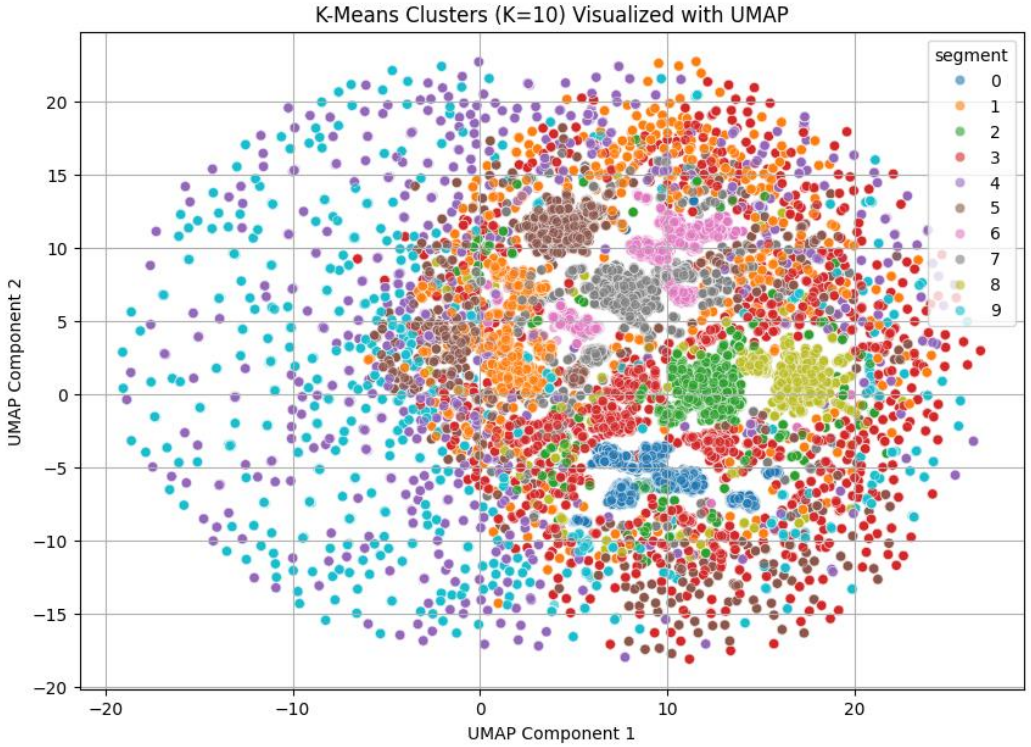
K-Means Clustering and Evaluation

The **K-Means clustering algorithm** was then applied to the PCA-transformed data to segment the population into the chosen **10 clusters**.

The segmentation model was evaluated using the **Silhouette Score**. This metric measures how similar an object is to its own cluster compared to other clusters, providing a measure of cluster cohesion and separation. The resulting **Silhouette Score of 0.19** was considered a reasonably good result for a high-dimensional and complex real-world dataset, indicating a fair degree of distinction between the identified segments.

Sample Cluster Interpretation

The segmentation model, utilizing K-Means on the PCA-transformed features, resulted in **10 distinct customer segments**. To visualize the separation and distribution of these clusters, a UMAP plot was generated .The initial sample interpretation focuses on two key dimensions: **Age** and **Net Gains** (the engineered feature combining capital gains, dividends, and losses), as these provide immediate insight into financial lifecycle and potential wealth.



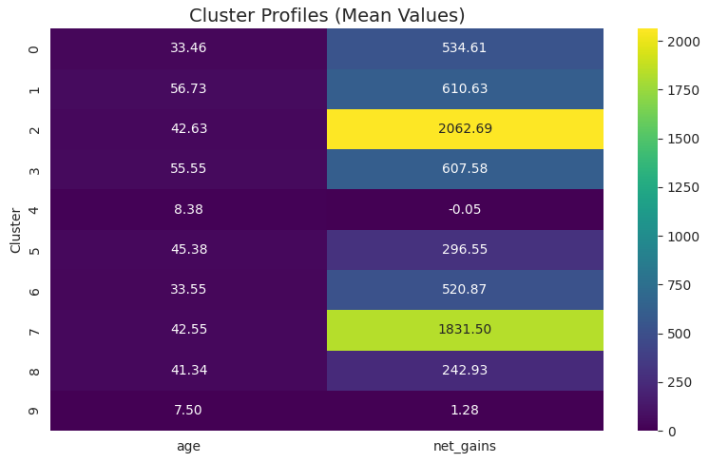
segment	count
9	27745
4	25243
3	23744
5	20788
0	19674
6	19528
7	16568
2	16408
1	16193
8	13631

Deeper Segment Profiling for Marketing

The true value of this segmentation model lies in its ability to facilitate further manual, in-depth analysis. For any given segment (e.g., the high-value Cluster 2), we can perform detailed profiling based on the full feature set:

- **Demographics:** Analyze gender (male-to-female ratio), education levels, race, and origin for culturally relevant messaging.
- **Employment:** Determine the distribution of occupation and industry to target advertising for professional services or industry-specific products.

Age and Net Gains Analysis



Segment Groups	Clusters	Average Age Range	Average Net Gains	Marketing Strategy Focus
Youngest/Developing	**9 and 4**	7 - 8	Low/Negative	Future potential, savings, or parent-focused products.
Core Mid-Range	**0, 5, 6, 8**	33 - 45	$242 - $534	Average segment. Focus on upselling and cross-selling everyday and mid-tier products.
High-Value/Wealthy	**2 and 7**	42 – 43	$1,831 -$2,062	**Most valuable.** High priority for retention, premium products, and growth strategies.
Established/Solid	**1 and 3**	Oldest Range	Intermediate	Solid base; target with mature products, stability, and retirement planning services.

Refining Cluster Granularity

Note that some clusters, such as Cluster 2 and 7 (similar mean age) or Cluster 1 and 3, appear to be closely related. While the number of clusters could be reduced to simplify the marketing effort, the current setting of **10 clusters** provides a higher degree of **granularity**, allowing for more specialized and potentially effective targeting strategies.

The resulting marketing strategies would thus move beyond simple income-based targeting to a multi-dimensional, segment-specific approach. The outputs of this model are designed to be used for this precise strategic execution.

Conclusion

This project successfully delivered a dual machine learning framework designed to optimize the retail client's marketing efforts.

1. Classification for Immediate Targeting

The supervised **Classifier** is an essential tool for **proactive income screening**.

- **Value:** The model scores are used to instantly predict if they are likely to earn <$50K or >$50K.
- **Action:** This enables the automated allocation of individuals to the correct high-tier or low-tier marketing funnel.

2. Segmentation for Strategic Campaign Design

The unsupervised **segmentation model** identified 10 distinct customer segments.

- **Value:** The segments reveal deeper customer profiles.
- **Action:** This enables personalized campaign development, allowing the client to tailor product offers, messaging tone, and channel strategy for each specific group, maximizing long-term engagement and marketing ROI.

Hardware and Environment

The project was executed using **Google Colab Pro**, including a **High RAM CPU** runtime environment to handle the large dataset and the iterative training required for the five classification models and the extensive hyperparameter search. All code was developed and documented within **Python notebooks**. It would be easy to show outputs, comments and headings in Notebooks.

Improvements and Future Work

To enhance the robustness, interpretability and production readiness

General Improvements

1. Modular code and Reusability: If local compute resources were not a limiting factor, the current notebook-based design could be refactored into modular design. Such as, creating separate python modules for various tasks (data loading, preprocessing, training, evaluation etc), writing reusable helper functions with descriptions, implementing flow diagrams, etc.
2. Model Explainability (XAI): We can incorporate XAI frameworks such as SHAP to move beyond basic feature importance and get instance level understanding.
3. Establish MLOps Lifecycle: We can implement an end-to-end MLOps pipeline that includes using tools like MLflow for model management (tracking experiments, parameters, metrics, challenger-champion model selection, inference, etc) setting up automated training and deployment pipelines, and managing a full model lifecycle.
4. Model Monitoring and Maintenance: We can develop monitoring dashboards to track model and data health in production. This includes detecting data drift or model drift to ensure model remain relevant
5. Real time inference

Classification Specific

1. Metric Optimization: While current model performance is acceptable, future work should focus on fine-tuning hyperparameters and utilizing advanced sampling techniques to improve minority class metrics. An inclined modeling approach toward Neural Network should also be studied.

Segmentation Specific

1. Model Robustness Testing: While in the experiments section DBSCAN was also tested, additional clustering algorithms should also be trained and compared.

References

1. Customer Segmentation using K-Means Clustering
2. End to End ML Classification Project & Algorithms
3. Gemini: Report Writing