JPMC Take Home Project

Author: Jin Young Choi

Date: Dec 15, 2025

Introduction

In this take home project, I was tasked with analyzing the 1994 and 1995 US census data to develop two models:

1. A binary classification model to differentiate between people who earn an income of less than $50,000 and more than $50,000.

2. A segmentation model to identify different groups of people for marketing purposes.

Note: Large language model (ChatGPT) was used to speed up coding and to generate the README.txt documentation. However, the decision-making process and the writing of this project report were done entirely by me.

Data Preprocessing

The initial phase was to probe into the dataset to get a general understanding of its format and guide the next step in the process. The codes and raw output for this section are provided in 0_data_exploration.ipynb.

Looking at the weighted distribution of income, we see that 93.8% of the people have an income of less than 50k, while only 6.2% has greater than 50k income. This seems a bit small by today's standards but after adjusting for inflation and buying power, 50k in 1995 is approximately $100k today [1].

Out of the 40 employment and demographic variables, I anticipated that some will be more relevant than others. Before applying any statistical techniques, I wanted to prune the features manually using intuition.

First, I looked at "duplicate" variables:

- "detailed industry recode" vs "major industry code"
- "detailed occupation recode" vs "major occupation code"

For our objective, I believe keeping simple, higher quality data would be better, so I removed the detailed recodes while keeping the major codes, which should have less sparsity as well.

I checked for variables with lot of missing entries. The four migration variables stood out with nearly 50% of its entries missing:

- “migration prev res in sunbelt”
- “migration code-change in msa”
- “migration code-change in reg”
- “migration code-move within reg”

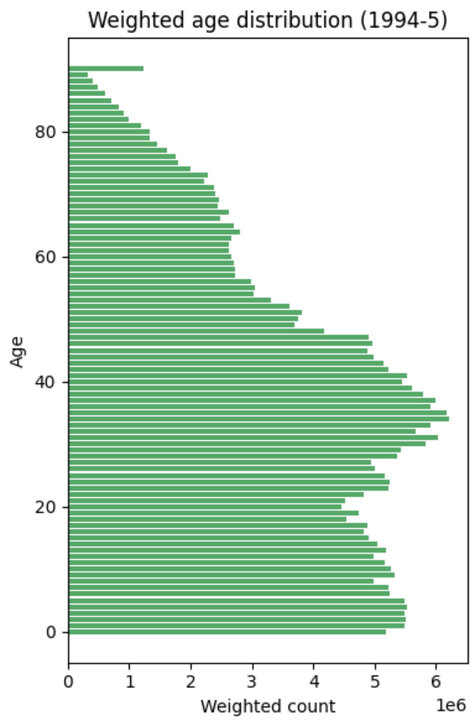
Similarly, some nearly constant variables with more than 95% of its entries being same were taken out:

- “fill inc questionnaire for veteran's admin”
- “reason for unemployment”

However, some nearly constant variables were kept (despite being almost all 0) as these relate to investments and likely closely tied to income.

- “capital losses”
- “capital gains”

An interesting side data is the distribution of the population pyramid compared to today. We can see a large influx of new babies (millennials) compared to the population decline observed today.



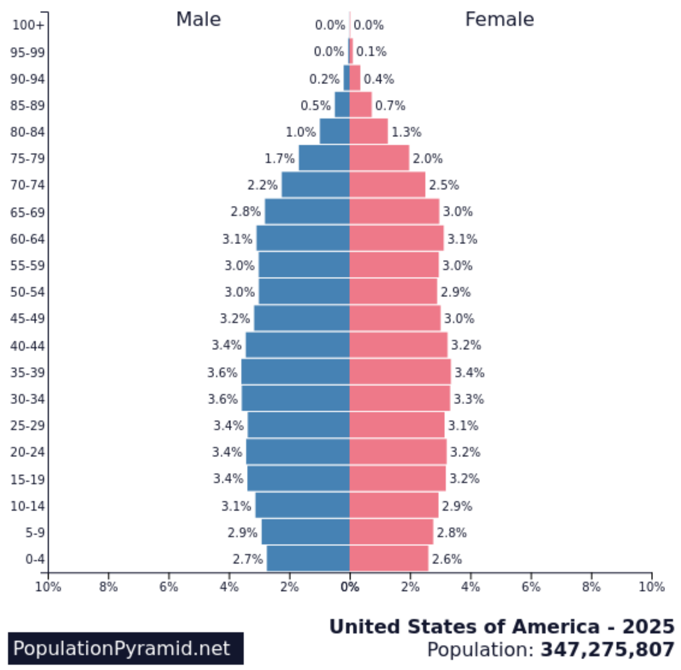


Figure 1. Age distribution of the 1994–1995 census sample (left) compared to the 2025 population pyramid (right) [2], highlighting a bulge of younger cohorts in the historical data versus a declining base in 2025

In addition to initial feature selection, the data is cleaned up to be used for training. "?" entries are treated as missing, string whitespace is stripped to avoid duplicate categories, and the income labels are encoded as 0 for < 50k and 1 for > 50k. The cleaned dataset is saved within the /preprocessed directory.

Objective 1: Income Classification

The code for the classifier model can be found in 2_classification.py. Further data processing occurs here through a simple median imputer to fill in missing values with its median for that variable, and categorical data are transformed into numerical features. The train-test data is randomly split 80-20 ratio. Common but effective models for binary classification logistic regression and random forest are trained and compared.

Logistic Regression

I used base logistic regression built into the scikit-learn library, as a good baseline for lightweight, interpretable model for binary income prediction. I tested three different hyperparameters $C \in \{0.5, 1.0, 2.0\}$, where C is the inverse of the regularization strength (i.e. smaller C means stronger L2 regularization, resulting in simpler weights with less overfitting and vice versa.)

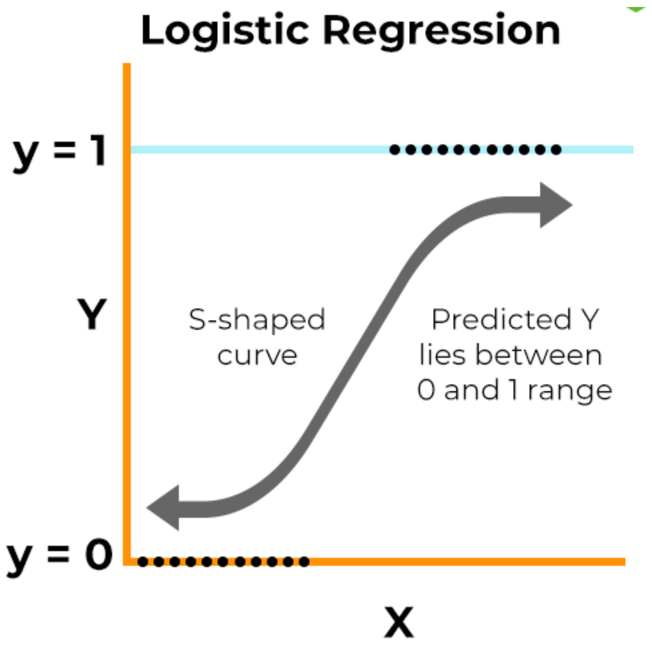


Figure 2. Schematic of logistic regression for binary classification [3]

Random Forest

Random forest takes an ensemble of decision trees to capture non-linear interactions, proven to be extremely effective on tabular data even compared to deep learning models [4]. I tested two different set of hyperparameters for number of trees and max depth of each tree (n_estimators=200 with max_depth=10, and n_estimators=200 with max_depth=None)

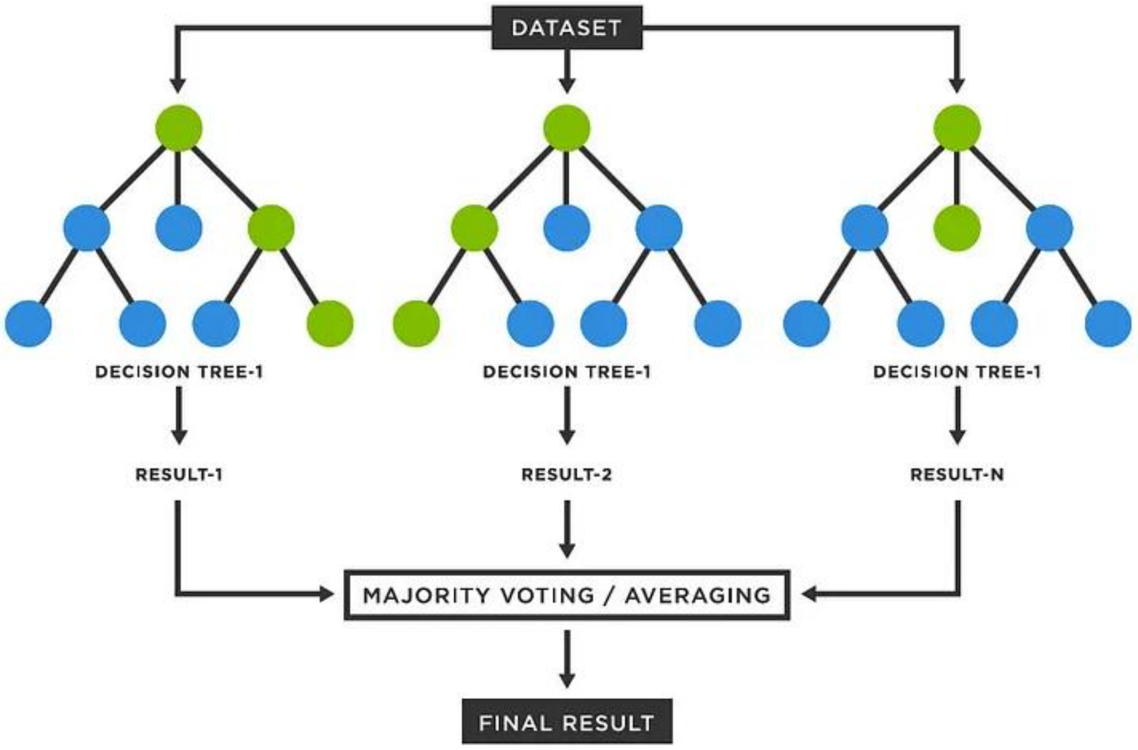


Figure 3. Schematic of random forest consisting of multiple decision trees to make the final prediction [5].

Results and Discussion

To evaluate, we can look at different metrics computed from the confusion matrix, consisting of true positive (TP), true negative (TN), false positive (FP), and false negative (FN). Due to the large class imbalance between <50k and >50K income (93.8% to 6.2%), accuracy (TP+TN)/Total can be misleading. For all the models tested, the accuracy was higher than 94%.

A better comparison metric would be positive-class precision (TP/(TP+N)), which shows that of the predicted >50k, how many are truly 50k. Also, positive-class recall (TP/(TP+FN)) shows, of the true >50k, how many we were able to catch, and F1 balances precision and recall for a single metric.

Figure 4. shows the comparison of the precision of the tested models. Interestingly, all the model perform similarly around ~0.73 but the random forest model (n_estimators=200, max_depth=10) performed much better at 0.93. However, the same random forest model performed the worst by a large margin in recall (Figure 5). This means that the model is very picky about classifying someone as >50k income, though it is usually correct when it does.

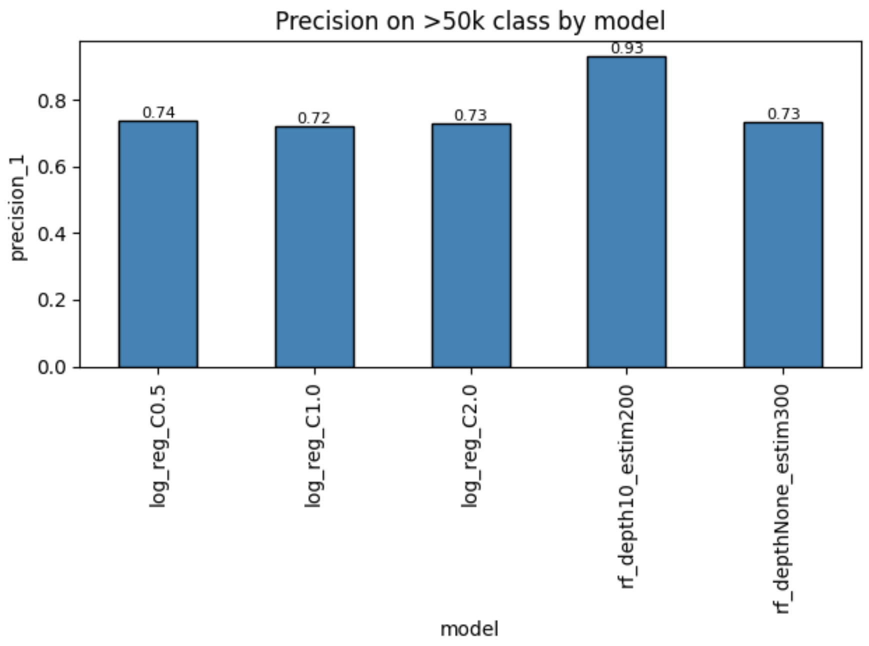


Figure 4. Positive-class precision comparison across models.

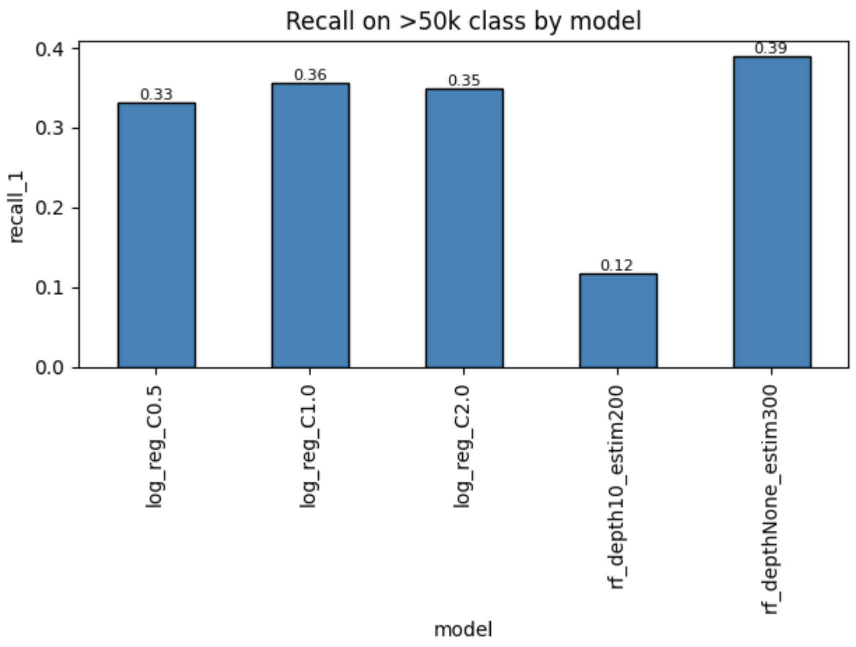


Figure 5. Comparison of positive class recall across models.

Examining the balanced F1 score (Figure 6) shows the best model as the random forest (n_estimators=300, max_depth=None) with an F1 score of 0.39, and the best logistic model (C=1.0) with an F1 score of 0.36. However, the logistic regression model is much more lightweight, as the random forest model stores hundreds of trees. I recommend the logistic regression model as it is fast to train and fast at inference, but if heavier computation is acceptable for small boost in recall, then the random forest model is recommended.

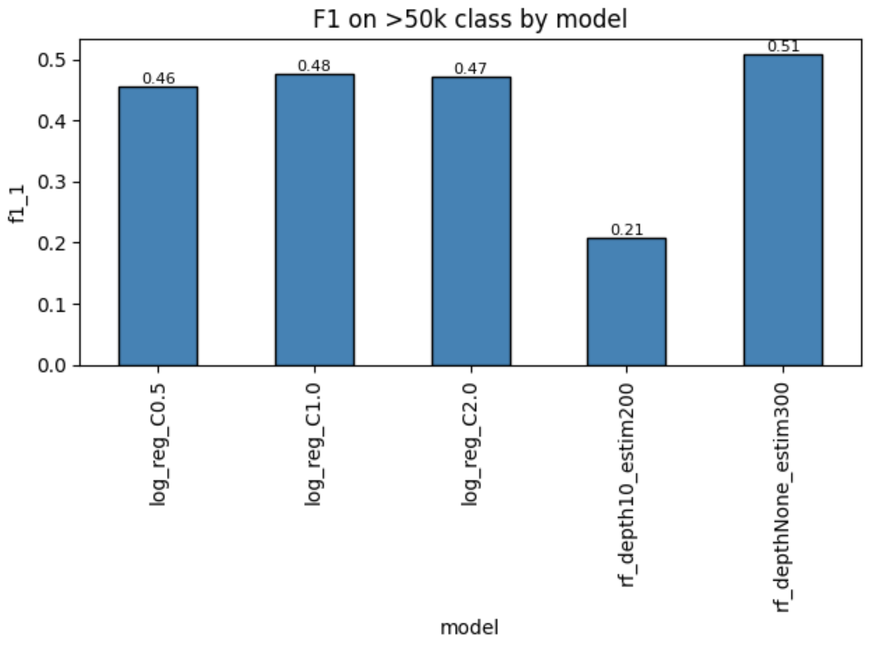
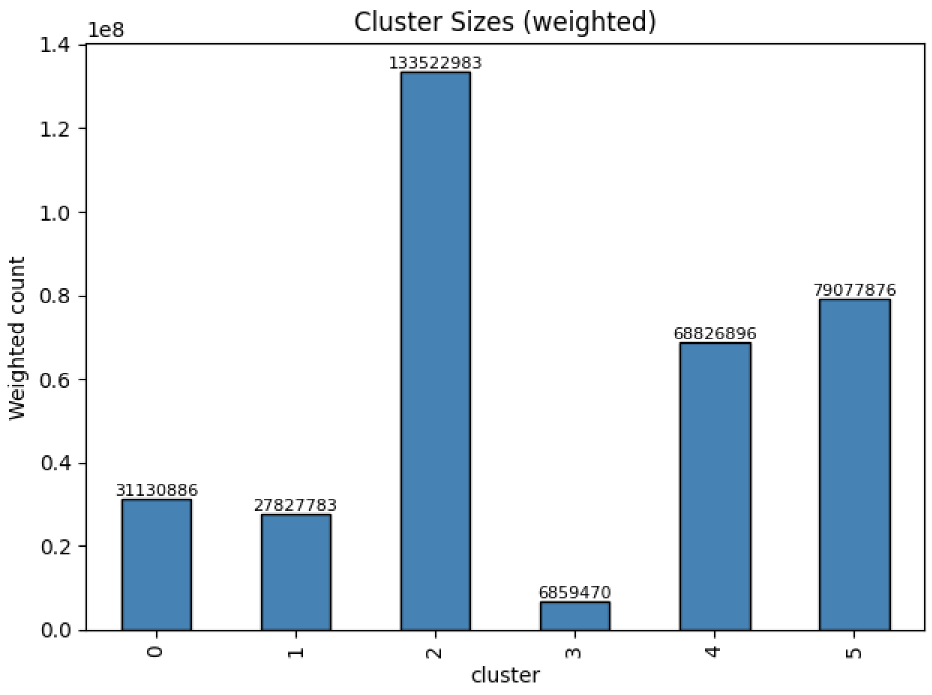


Figure 6. Comparison of positive-class F1 across models.

Objective 2: Segmentation

For the segmentation model, I applied an unsupervised model, K-means clustering due to its speed and scalability on a large feature matrix, and relatively simple and interpretable outputs [6]. After testing multiple number of clusters, I decided on a balance of 6 clusters, which differentiated between broad customer groups with different income levels without over- or under-clustering. The weight cluster sizes, and share of >50k income for each cluster is shown in Figure 7.



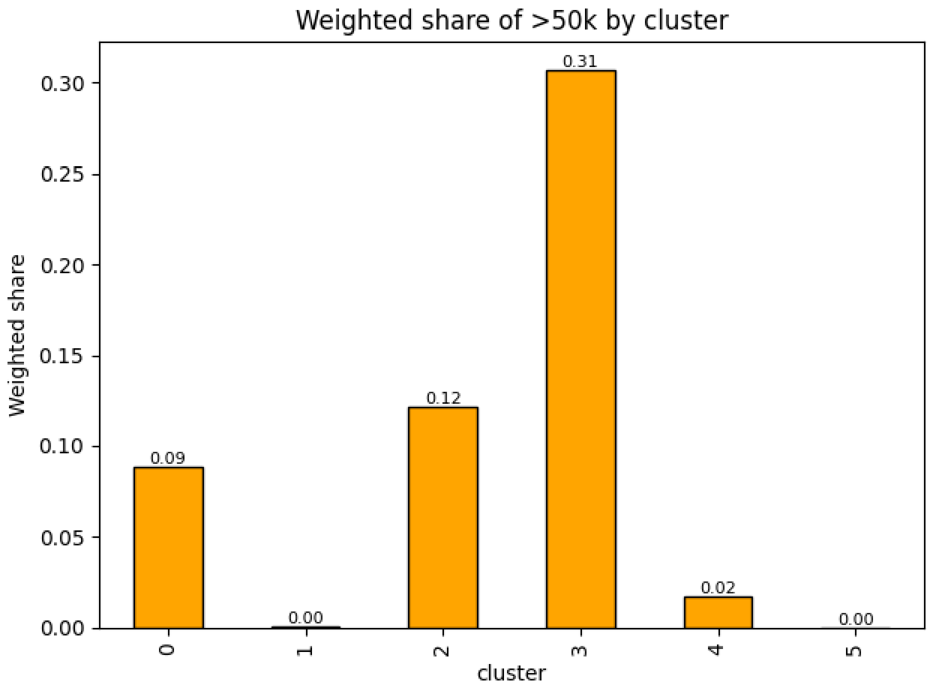


Figure 7. Weight cluster sizes for 6 different clusters, and share of >50k income for each cluster.

Cluster 3 has the lowest number of people but the highest fraction of people with >50k income. This can be seen as the high-value group. I recommend prioritizing premium offers to this group, perhaps through personalized channels such as mail or email.

Cluster 0 and 2 comprise of a larger group of people with mid-level fraction of people with high income. They can be seen as a mid-value group, to whom we can offer mid-level products, potentially through mainstream channels such as TV or online advertisements. For more detailed marketing, we can parse through the difference between the two clusters to craft different messaging to match their demographics. For example, quick comparison shows that cluster 0 tends to be slightly younger and unmarried, while cluster 2 is older and more family-oriented.

Cluster 1, 4, and 5 also comprise a large group of people but with low income. I recommend offering low-cost, and budget products and avoid more premium options. Again, the three groups can be profiled to optionally refine the messaging.

Conclusion/Future Works

In summary, I trained a weighted income classifier (<50k vs >50k) with manual feature pruning and one-hot preprocessing. Logistic regression and random forest models are tested with basic hyperparameter tuning, and the random forest model (n_estimators=300) delivered the best overall F1 (0.51) metric, though at a larger training and inference cost. The logistic regression (C=1) had a slightly lower (F1=0.48) but similar performance at a much lighter computational cost. A future work can be trying gradient boostin (e.g. XGBoost) and feature engineering to improve recall on the >50k class.

Segmentation with K-means clustering (K=6) produced a small high-value cluster (~30% $50k), mid-value clusters (~9–12% >50k), and low-value clusters (~0–2% >50k). This can help guide marketing decisions, such as premium offers to the high-value niche, mid-tier offers to medium clusters, and budget options for low-value clusters. Further demographic analysis between the clusters can be performed in the future to create more personalized marketing campaigns.

References

[1] Inflation Calculator. Available: https://www.in2013dollars.com/us/inflation/1995?amount=50000

[2] “Population of United States of America 2025,” PopulationPyramid.net. Available: https://www.populationpyramid.net/united-states-of-america/2025/

[3] V. Kanade, “What Is Logistic Regression? Equation, Assumptions, Types, and Best Practices,” Available: https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/

[4] L. Grinsztajn, E. Oyallon, and G. Varoquaux, “Why do tree-based models still outperform deep learning on tabular data?,” arXiv:2207.08815 Available: https://arxiv.org/abs/2207.08815

[5] “Understanding Random Forest Algorithm: A Comprehensive Guide,” Available: https://datasciencedojo.com/blog/random-forest-algorithm/

[6] A. Kumar, “A Simple Explanation of K-Means Clustering,” Available: https://www.analyticsvidhya.com/blog/2020/10/a-simple-explanation-of-k-means-clustering/