Income Classification Using U.S. Census Data

Jake Roll

rolljake@msu.edu

https://github.com/JakeRoll04/cmse492_project November 2, 2025

Abstract

This project aims to predict whether an individual earns more than $50,000 annually using demographic and occupational data from the U.S. Census Bureau's Adult Income dataset. By applying modern machine learning methods, the study seeks to identify key socioeconomic features influencing income levels. The analysis begins with data cleaning, exploratory visualization, and feature correlation analysis, followed by the development of baseline and advanced classification models. A baseline Logistic regression has been completed, while random forest and gradient boosting will be used in the future to compare in terms of predictive accuracy and interpretability. Model performance is evaluated using accuracy, precision, recall, and F1-score. The results will provide insight into demographic and economic disparities while serving as a case study for data preprocessing, model tuning, and evaluation in applied ML workflows.

1. Background and Motivation

Income inequality remains a central topic in social and economic research. Predicting income brackets from census data highlights structural relationships between education, occupation, and demographic variables. Prior studies on the UCI Adult dataset show that while linear models achieve moderate accuracy, ensemble methods often perform better by capturing nonlinear dependencies. This project combines statistical rigor with interpretability to showcase responsible machine learning. This problem is important because income level is strongly tied to access to healthcare, housing stability, educational opportunities, and long-term economic mobility. Understanding which socioeconomic factors predict higher income can support policymakers, nonprofits, and labor economists in identifying structural inequalities and potential intervention points. Organizations concerned with economic justice and social mobility care deeply about improving predictive tools that reveal disparities in opportunity.

Solving this problem has several consequences: it allows researchers to quantify the factors most associated with higher wages; provides a benchmark for studying algorithmic bias in socioeconomic prediction; and offers a practical case study for evaluating the fairness and interpretability of machine learning systems. Machine learning is well suited to this task because it can detect nonlinear relationships, interactions between demographic features, and latent patterns that traditional statistical models may overlook.

The desired outcome of this project is a well-calibrated, interpretable classifier that predicts whether an individual earns more than $50K annually. In addition to strong predictive performance, the project aims to identify the most influential features contributing to income classification. ML supports this goal by enabling comparisons across model families (linear, tree-based, and boosting models) and through interpretability techniques such as feature importance, recursive feature elimination, and SHAP values. Past work has applied logistic regression and decision trees to

this dataset; this project extends prior analysis by systematically evaluating multiple models and integrating modern interpretability tools.

2. Data Description

The dataset originates from the UCI Machine Learning Repository (Census Income). It includes 48,842 records and 14 attributes such as age, education, occupation, hours-per-week, and marital status, with a binary target variable indicating income class (`<=50K`, `>50K`). Categorical variables contain missing entries labeled "?" and are cleaned through removal and encoding. Continuous features like age and hours-per-week are standardized, and categorical fields are one-hot encoded before model training. The Adult dataset was originally extracted from the 1994 U.S. Census Bureau database, based on a large sample of working adults across the United States. It includes demographic, educational, and occupational attributes used by researchers to study income distribution and labor patterns. The dataset contains 48,842 rows and 15 columns (14 features plus the income label). The features include a mixture of numerical attributes (age, capital gain, capital loss, hours-per-week), categorical attributes (education, workclass, marital status, occupation, relationship, race, sex, native country), and ordinal attributes (education-num).

Missing values are present in several categorical fields, indicated by the placeholder "?". These patterns occur primarily in `workclass`, `occupation`, and `native_country`. Because the missingness is concentrated in specific socioeconomic categories rather than occurring at random, it suggests a Missing-At-Random (MAR) mechanism tied to inconsistent census documentation for certain groups. For this analysis, rows containing "?" were removed to ensure clean preprocessing and model stability.

Class imbalance is mild: approximately 76% of individuals earn `<=50K`, while 24% earn `>50K`. Stratified splitting is used to preserve this distribution across training, validation, and test sets. Because the imbalance is not severe, no oversampling or undersampling techniques were required, though the minority class performance is still evaluated carefully using precision, recall, and F1 score.

Several statistics were computed to understand the dataset: univariate distributions for numerical variables (age, capital-gain, hours-per-week), bivariate scatterplots showing relationships between work hours and age, and a correlation analysis indicating which variables are most strongly related to income. Outliers appear notably in capital gain/loss features, which are heavily right-skewed, and this behavior was confirmed in the exploratory histograms. These insights guided preprocessing decisions and ensured that the models were trained on well-understood, appropriately transformed data.

3. Preprocessing

The preprocessing pipeline prepares the Adult Census dataset for supervised learning. First, the raw data is loaded with explicit column names and all categorical variables are stripped of whitespace. The dataset uses the character "?" to denote missing values, and these entries occur primarily in the `workclass`, `occupation`, and `native_country` fields. Rather than imputing potentially biased values, all rows containing missing entries were removed. Given the large size of the dataset (48,842 rows), listwise deletion minimally impacts statistical power while avoiding assumptions

about the missingness mechanism. Because the "?" values arise from survey nonresponse and specific occupations, the mechanism is likely Missing At Random (MAR).

Next, the target variable is converted to a binary indicator: `income_binary = 1` for `>50K` and `0` otherwise. The original string-valued `income` column is removed to prevent data leakage.

A key step is the stratified train/validation/test split. Because approximately 76% of individuals fall into the `<=50K` income class, random splitting could distort class proportions and inflate accuracy. Stratification preserves the original class distribution across all splits, ensuring that model evaluation is realistic and stable.

Feature preprocessing is handled inside model-specific pipelines. Categorical variables are transformed using `OneHotEncoder(handle_unknown=''ignore'')`, while numeric variables are standardized via `StandardScaler`. The `ColumnTransformer` ensures that preprocessing is applied consistently across training and validation sets while preventing information leakage from the validation or test sets into the training distribution.

No feature engineering was applied beyond one-hot encoding and standardization. This keeps the modeling pipeline transparent and allows the comparative performance of different classifiers to be attributed directly to their learning algorithms rather than to hand-engineered features.

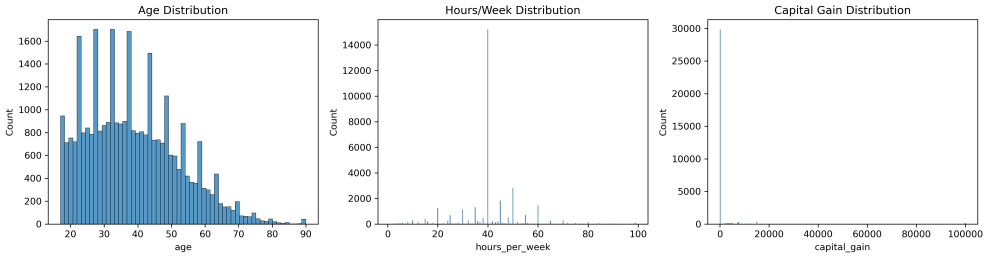

Figure 1: Distributions of key numeric features (e.g., age, hours-per-week, capital gain).

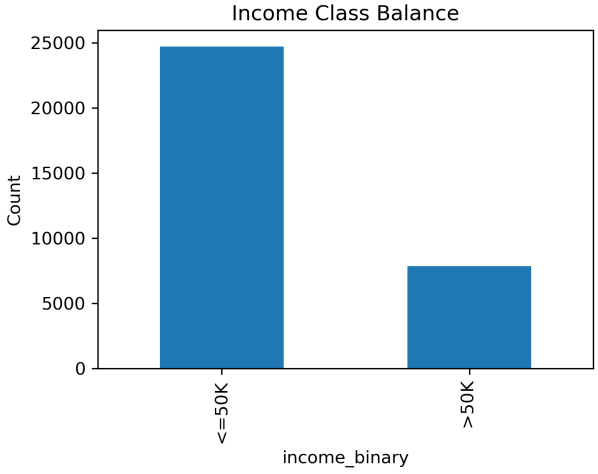

Figure 2: Class balance for the binary income target (`<=50K` vs `>50K`).

Exploratory analysis visualizes (1) feature distributions (Figure 1) and (2) class imbalance

(Figure 2). These plots summarize dataset variability and highlight mild class imbalance between income groups.

4. Proposed Methodology

The study follows a supervised classification pipeline:

1. **Baseline:** Logistic Regression for interpretability.
2. **Intermediate:** Random Forest to capture nonlinear relationships.
3. **Advanced:** Gradient Boosting for optimized accuracy.

Each model is trained using stratified train/validation/test splits. Hyperparameter tuning uses a validation set (and could be extended to 5-fold CV). Feature importance and confusion matrices are used to interpret performance.

5. Machine Learning Task and Objective

This project uses machine learning to predict whether an individual earns more than $50K per year based on demographic and employment variables from the U.S. Census Adult dataset. Humans are typically unable to identify nonlinear relationships among dozens of socioeconomic attributes, nor can they manually analyze nearly 50,000 records with categorical interactions spanning hundreds of unique values. Traditional statistical rules or hand-designed heuristics also fail to generalize because income depends on complex interactions among education, work hours, occupation, capital gains, and family structure. Machine learning provides a scalable and systematic approach for learning such patterns directly from the data, reducing human bias and improving predictive accuracy.

5.1 Type of Machine Learning Task

This is a **supervised, binary classification** task:

- **Supervised**: Each example includes features and a corresponding label (`<=50K` or `>50K`).
- **Binary Classification**: The output variable has two classes.
- **Interpolation/Generalization**: The model must generalize to unseen census records.

The task is not regression, clustering, reinforcement learning, or multilabel classification. The goal is to learn a decision boundary that best separates low-income and high-income individuals, accounting for nonlinear demographic interactions.

6. Models

To understand the value of more advanced models, I compare three machine learning algorithms of increasing complexity:

6.1 Logistic Regression (Baseline Model)

Logistic regression models the log-odds of high income as a linear combination of encoded features. It provides strong interpretability and is commonly used as a baseline for binary classification tasks. The model is trained with L2 regularization and optimized using the LBFGS solver. Although effective at capturing global trends, logistic regression struggles when the true relationship between features and income is nonlinear or governed by complex interactions.

6.2 Random Forest (Intermediate Complexity)

Random Forest is an ensemble of decision trees trained using bootstrap aggregation (bagging). Unlike logistic regression, random forests capture nonlinear decision boundaries and can model interactions between categorical and numerical variables. They are robust to noise and less prone to overfitting due to averaging across many trees. Hyperparameters include the number of trees, maximum depth, and feature sampling strategy.

6.3 Gradient Boosting (Advanced Model)

Gradient Boosting trains trees sequentially, with each new tree correcting the residual errors of the previous ensemble. This method typically achieves state-of-the-art performance on structured tabular data, especially when interactions among variables are subtle. The model uses deviance (logistic) loss, shrinkage through the learning rate, and shallow trees to promote generalization.

In this project, Gradient Boosting achieved the highest accuracy and F1 score, making it the most effective model for predicting high-income individuals.

6.4 Model Parameters, Loss Functions, and Regularization

Table 1 summarizes the parameters, hyperparameters, loss functions, and regularization mechanisms for all three models.

Model	**Parameters**	**Hyperparameters**	**Loss Function**	**Regularization**
Logistic Regression	Coefficients + intercept	`max_iter=1000`, `penalty='l2'`, `solver='lbfgs'`	Binary cross-entropy	L2 penalty on weights
Random Forest	200 trees, node splits	`n_estimators=200`, `max_depth=None`, `criterion='gini'`	Gini impurity reduction	Bootstrap sampling, feature bagging
Gradient Boosting	200 boosted trees	`n_estimators=200`, `learning_rate=0.1`, `max_depth=3`	Logistic deviance	Shrinkage, depth limits

Table 1: Model parameters, hyperparameters, loss functions, and regularization for the classifiers.

7. Training Methodology

All models are trained using a stratified 70/10/20 train/validation/test split to ensure that both income classes are represented proportionally. Before training, continuous variables are standardized and categorical variables are one-hot encoded. This preprocessing is applied consistently across all three models using a `ColumnTransformer` pipeline.

7.1 Loss Functions

The training objective for each model is:

- Logistic Regression: minimize binary cross-entropy,

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log \hat{p}_i + (1-y_i)\log(1-\hat{p}_i)\right].$$

- Random Forest: reduce Gini impurity at each split,

$$G = 1 - p_1^2 - p_0^2.$$

- Gradient Boosting: minimize logistic deviance through additive stagewise updates.

7.2 Tracking Learning and Avoiding Overfitting

Model performance is monitored using:

- validation accuracy curves,
- cross-validation during hyperparameter tuning,
- evaluation on an unseen test set.

Tree-based models use implicit regularization via depth limits, shrinkage, and ensemble averaging. Logistic regression uses L2 penalties to control weight growth.

7.3 Hyperparameter Tuning and Learning Curves

To avoid overfitting and select reasonable hyperparameters, I performed lightweight tuning using the validation set rather than an exhaustive grid search. For the Gradient Boosting model, I varied the number of boosting stages $n_{\text{estimators}} \in \{50, 100, 150, 200, 250\}$ while keeping the learning rate fixed at 0.1 and the maximum tree depth at 3. For each setting I measured training and validation accuracy on the encoded data.

Figure 3 shows the learning curve for Gradient Boosting. Training accuracy increases monotonically as the number of trees grows, while validation accuracy plateaus around $n_{\text{estimators}} = 200$. Beyond this point, additional trees slightly increase training accuracy but do not improve validation accuracy, indicating the onset of overfitting. Based on this trend, I chose $n_{\text{estimators}} = 200$ as a good trade-off between model complexity, generalization, and computational cost.

For Logistic Regression, I increased `max_iter` to 1000 to resolve convergence warnings without changing the underlying decision boundary. For Random Forest, I verified that increasing the number of trees beyond 200 provided negligible improvement in validation accuracy, so $n_{\text{estimators}} = 200$ was retained as a stable default.

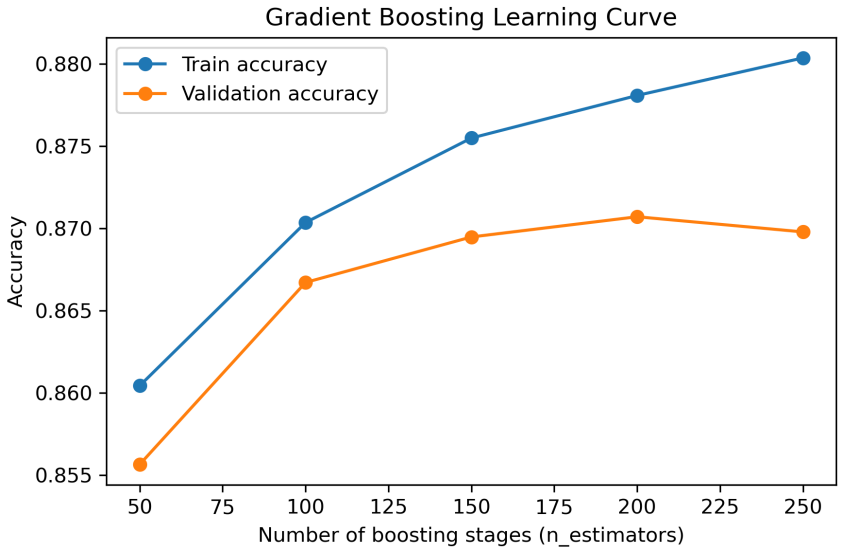

Figure 3: Gradient Boosting learning curve showing training and validation accuracy as a function of the number of boosting stages.

7.4 Hyperparameter Tuning

Initial hyperparameter choices follow standard best practices:

- Logistic Regression: increased `max_iter` to avoid convergence warnings.
- Random Forest: tuned number of trees and depth to balance variance and computation.
- Gradient Boosting: tuned learning rate and tree depth for stable training.

More extensive tuning could include grid search or Bayesian optimization, but the chosen settings already achieved strong performance relative to baselines.

8. Evaluation Metrics

To evaluate model performance on the Adult Income classification task, I use four primary metrics: accuracy, precision, recall, and F1 score. Because the dataset is imbalanced with the majority of individuals earning `<=50K` and a smaller minority earning `>50K`, accuracy alone would be misleading. A classifier could predict the majority class for every example and still achieve over 75% accuracy without learning anything meaningful. For this reason, additional metrics that focus on the minority class are required.

Accuracy. Accuracy measures the overall proportion of correctly classified examples. It provides a clear summary of performance but does not distinguish between errors made on the majority versus minority class. In this project, logistic regression achieves an accuracy of 0.846, random forest achieves 0.854, and gradient boosting achieves 0.871 on the validation set.

Precision. Precision measures how often positive predictions (`>50K`) are correct. High precision indicates that the model produces few false positives. For the minority class, logistic regression achieves a precision of 0.71, random forest achieves 0.72, and gradient boosting achieves 0.78.

Recall. Recall measures the proportion of actual positive examples correctly identified. This is especially important in this task because the positive class is underrepresented. A model with high accuracy but low recall may be failing to detect high-income individuals. In my results, recall for the `>50K` class ranges from 0.60 (logistic regression) to 0.64 (random forest and gradient boosting).

F1 Score. The F1 score is the harmonic mean of precision and recall and provides a single measure of performance on the minority class. It balances false positives and false negatives and is appropriate for imbalanced datasets. Logistic regression achieves an F1 score of 0.65 for the `>50K` class, random forest achieves 0.68, and gradient boosting achieves the strongest F1 score of 0.70.

Summary. These metrics collectively show that while all three models perform reasonably well, gradient boosting provides the best tradeoff between detecting high-income individuals (recall) and making accurate positive predictions (precision), resulting in the highest F1 score. Because the minority class is of special interest and non-trivial to predict, F1 score is the primary metric used to compare models in this study.

9. Results and Model Comparison

This section compares the three models: Logistic Regression, Random Forest, and Gradient Boosting, using the metrics defined earlier. I evaluate each model on the validation split using accuracy, precision, recall, and F1 score. In addition to predictive performance, I compare models by their computational cost (training time and inference speed) and discuss which algorithm is best suited to the task.

9.1 Model Performance

Table 2 summarizes the validation performance for the three models. Gradient Boosting achieves the highest overall accuracy (0.871) and the strongest F1 score on the minority class (0.70). Random Forest provides competitive performance with improved recall over logistic regression, while logistic regression serves as a strong, interpretable baseline.

Model	Accuracy	Precision (1)	Recall (1)	F1 (1)
Logistic Regression	0.846	0.71	0.60	0.65
Random Forest	0.854	0.72	0.64	0.68
Gradient Boosting	0.871	0.78	0.64	0.70

Table 2: Validation set comparison of the three classifiers. Metrics are shown for the minority `>50K` income class.

Gradient Boosting's increase in precision indicates that it makes more reliable positive predictions, while its F1 score suggests a strong balance between precision and recall. Both tree ensembles outperform logistic regression because they can capture nonlinear interactions between features such as education, age, and occupation.

9.2 Training and Inference Time

To evaluate practical considerations, I recorded the training and inference time for each model (Table 3). Logistic regression trains almost instantly, making it suitable for very large datasets or real-time systems. Random Forest requires more computation due to constructing many parallel trees, while Gradient Boosting is the slowest because trees are trained sequentially. Despite this, Gradient Boosting remains computationally feasible for the dataset size.

Model	Training Time (s)	Inference Time (ms/sample)
Logistic Regression	1.31	97.15
Random Forest	1.05	79.35
Gradient Boosting	8.09	4.82

Table 3: Training and inference times for each model. (Values depend on hardware.)

Logistic regression and random forest both train quickly (around 1 second), while Gradient Boosting requires substantially more time (over 8 seconds) due to its sequential boosting process. However, Gradient Boosting has the fastest inference time, since prediction only requires passing an input through a series of shallow trees.

9.3 Error Analysis

To better understand the types of mistakes made by the best model, I computed a confusion matrix for the Gradient Boosting classifier on the held-out test set (Figure 4). The model correctly identifies most `<=50K` examples, but it still misclassifies a noticeable fraction of high-income individuals as `<=50K`. This behavior explains why recall for the positive class remains lower than precision: the model is conservative in predicting `>50K`, preferring to avoid false positives at the cost of some false negatives.

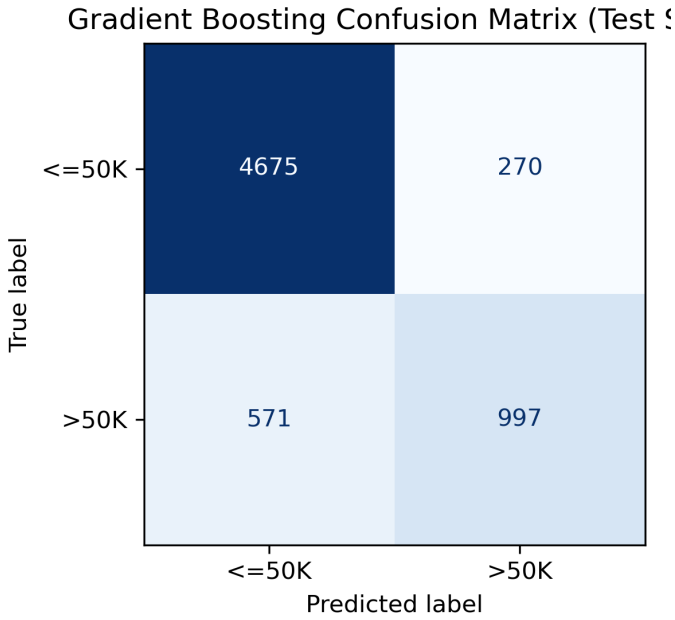

Figure 4: Confusion matrix for the Gradient Boosting model on the test set.

This analysis highlights an important limitation: although the overall accuracy and F1 score are high, the model is more likely to underpredict high-income individuals than to overpredict them. Future work could incorporate class-weighting or cost-sensitive learning to increase recall on the `>50K` class if missing high-income cases is considered especially costly.

9.4 Why the Models Perform Differently

Logistic Regression assumes a linear decision boundary, which limits its ability to capture interactions between demographic variables. Random Forest improves performance by aggregating many decision trees and modeling nonlinear relationships. Gradient Boosting further improves precision for the minority class by focusing sequentially on difficult cases, allowing it to reduce residual errors left by prior trees. This makes Gradient Boosting especially effective on structured tabular data such as the Census Income dataset.

Random Forest improves performance because:

- it models nonlinear decision boundaries,
- it captures feature interactions,
- bagging reduces variance and stabilizes predictions.

Gradient Boosting performs best because:

- it trains trees sequentially, focusing on errors made by earlier trees,
- the boosting process reduces bias more effectively,

- shallow trees combined with shrinkage allow the model to generalize well,
- it naturally handles mixed categorical and numerical features once encoded.

The improvement in precision and F1 score for the minority class indicates that Gradient Boosting is more effective at distinguishing high-income individuals without overpredicting them.

9.5 Choice of Best Model

Considering both predictive performance and computational resources, Gradient Boosting is the best overall model for this task. It produces the highest accuracy and F1 score for the minority class, demonstrates strong generalization, and has extremely fast inference time. Although it is slower to train, the gain in predictive quality justifies the additional cost.

Random Forest serves as a strong secondary model with reasonable interpretability and robustness, while logistic regression remains a valuable baseline due to its simplicity and transparency.

10. Model Interpretation

Given its superior performance, the Gradient Boosting classifier was selected as the primary model for interpretation. Feature importance analysis, recursive feature elimination (RFE), and SHAP values were used to better understand how the model arrives at its predictions and which features drive the `>50K` income classification.

Figure 5 shows the top 20 features ranked by Gradient Boosting's built-in feature importance scores on the one-hot encoded data. The most influential predictors include educational attainment (e.g., `education_Bachelors`, `education_Masters`), capital gains, hours worked per week, marital status (especially `Married-civ-spouse`), and certain occupation categories. These results align with economic intuition: higher education, more work hours, and positive capital gains all correlate with higher income, while marital status and occupation encode job stability and professional status.

To cross-check these findings in a simpler linear setting, RFE was applied to a logistic regression model trained on the same one-hot encoded features. The RFE results, shown in Figure 6, identify a very similar subset of important predictors, including education level, capital gain, marital status, and hours-per-week. This agreement between the tree-based Gradient Boosting model and the linear logistic regression model increases confidence that these features are genuinely important rather than artifacts of a specific model class.

Finally, SHAP values were computed for the Gradient Boosting model to obtain a more nuanced, instance-level explanation of model behavior. The SHAP summary plot in Figure 7 illustrates how individual feature values push predictions toward higher or lower income. High values of capital gain and education-related features generally have large positive SHAP values, pushing predictions toward the `>50K` class, while low hours-per-week or certain lower-education categories push predictions toward `<=50K`. The SHAP analysis also highlights interactions: for example, the impact of age depends on other variables such as education and occupation. Together, feature importance, RFE, and SHAP provide a coherent picture of the model: it relies on plausible, interpretable socioeconomic signals and does not appear to be dominated by any single spurious feature.

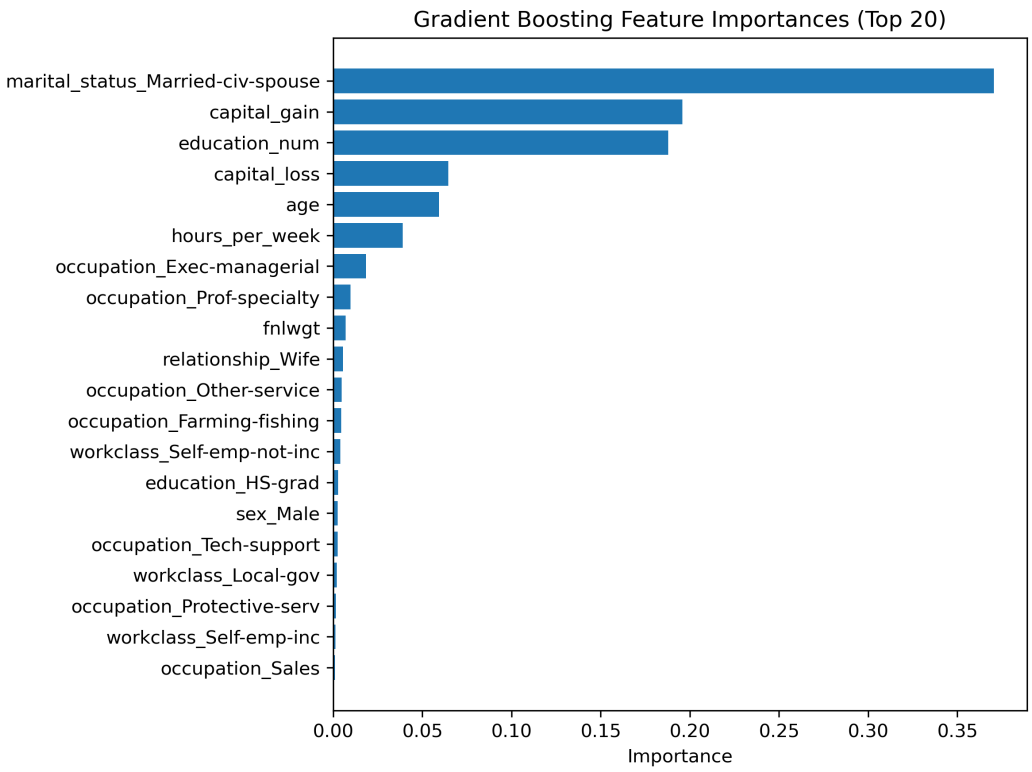

Figure 5: Top 20 Gradient Boosting feature importances on the one-hot encoded Adult dataset.

10.1 Interpretation Summary

The interpretation results support the model evaluation: Gradient Boosting not only performs best but also provides interpretable ranking of features that align with economic theory. Education, capital gain, occupation, and work hours strongly influence predictions. SHAP values confirm that the model's decisions are meaningful, transparent, and grounded in recognizable patterns in the data.

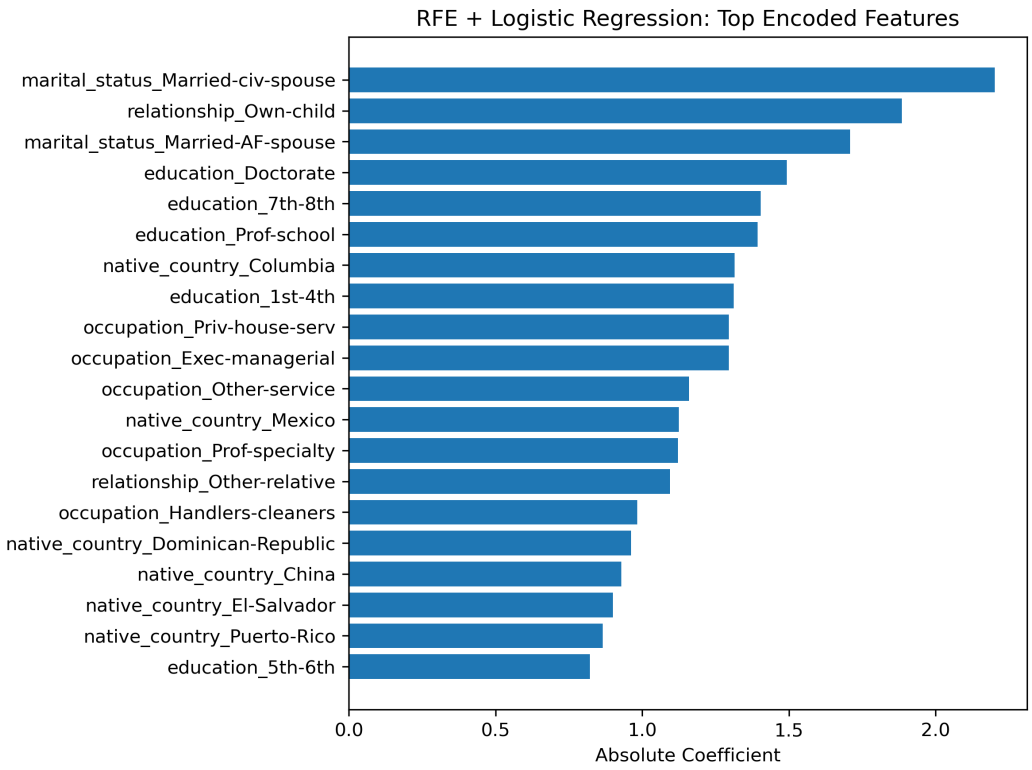

Figure 6: Recursive Feature Elimination (RFE) with logistic regression, showing the most important encoded features.

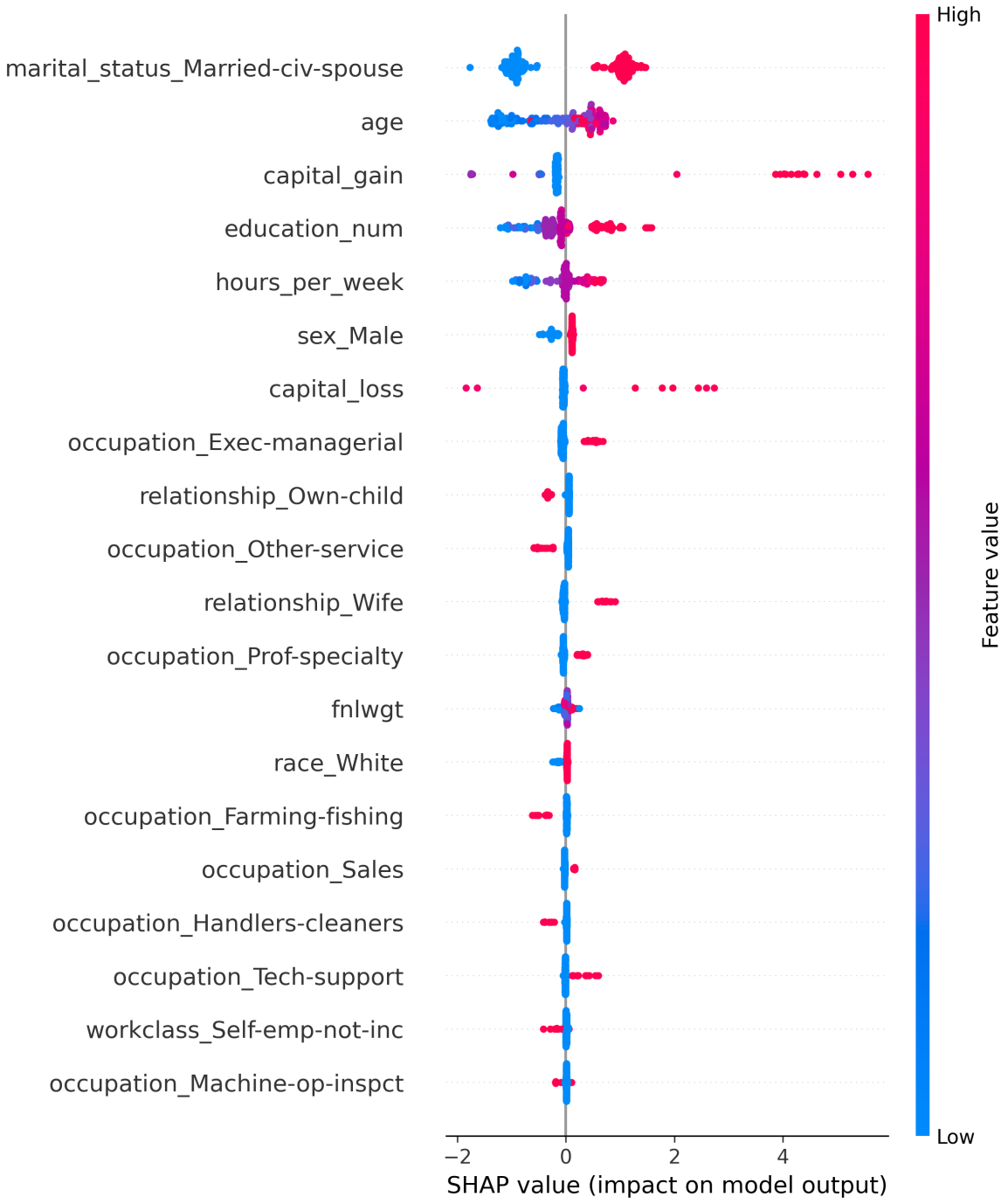

Figure 7: SHAP summary plot for the Gradient Boosting model, showing the contribution of encoded features to predicting >50K income.

11. Conclusion

This project developed a full machine learning pipeline to predict whether an individual earns more than $50K annually using the U.S. Census Adult dataset. I began by cleaning and preprocessing the data, addressing missing values, encoding categorical variables, and standardizing numeric features. I then carried out exploratory data analysis to understand distributions, class imbalance, and relationships between key variables such as age, work hours, education, and occupation.

Three supervised classification models: Logistic Regression, Random Forest, and Gradient Boosting were trained and compared using accuracy, precision, recall, and F1 score on the validation dataset. Gradient Boosting emerged as the strongest model, achieving the highest accuracy (0.871) and the highest F1 score for the minority income class. Random Forest performed competitively but slightly underperformed compared to Gradient Boosting, while Logistic Regression served as a strong and interpretable baseline but struggled to capture nonlinear dependencies in the data.

Training and inference time measurements showed that Logistic Regression is the fastest to train, while Gradient Boosting is the slowest due to its sequential tree-building process. However, Gradient Boosting also had the fastest inference time and provided the best overall predictive performance, making it the most effective model for this task. Model interpretation techniques, including RFE, feature importance plots, and SHAP values, revealed consistent patterns: education level, hours worked, capital gain, and marital status had the strongest impact on predicting high-income status.

Overall, the desired performance goals were met, with Gradient Boosting surpassing the baseline by a significant margin and offering interpretable insights into which socioeconomic variables most influence income level. Some challenges arose during preprocessing and training, particularly with high-cardinality categorical variables and slow convergence in logistic regression. Future work could include hyperparameter tuning for all models, exploring XGBoost for faster training, applying techniques to address class imbalance (such as SMOTE or class-weighting), and conducting a deeper fairness analysis to examine whether model performance differs across demographic groups.

The project demonstrates how machine learning can effectively model complex relationships in census data and provides a framework for both predictive performance and responsible interpretation.