Income Prediction Based On Census Data

Sourabh Shetye, R. K. Aadhithya, and G. Shreenivasan
Department of Computer Science
BITS Pilani, Dubai Campus
Sourabh.shetye- f20220231@dubai.bits-pilani.ac.in, RK.Aadhithya
f20220308@dubai.bits-pilani.ac.in, G.Shreenivasan f20220280@dubai.bits-pilani.ac.in

Abstract:

Income prediction is a major area of study, where demographic and socioeconomic attributes are utilized to forecast whether a person's annual income is higher or lower than some value (typically $50,000). This paper talks about the use of machine learning techniques on the UCI Adult Census dataset to forecast income levels. We discuss models such as Random Forest, SVM, and Deep Learning, and mention preprocessing, feature selection, and class imbalance techniques. The proposed methodology includes a robust pipeline with data cleaning, model training, and performance comparison using different metrics. We demonstrate that ensemble learners like Gradient Boosting are better than regular models and provide a fair, more accurate prediction system.

Keywords: Income prediction, Census data, Machine Learning, Feature selection, Class imbalance, Random Forest, SVM, Deep Learning.

1. Introduction

Economic, public policy, finance, and social studies research all stand to gain significantly from income prediction based on census data. The goal is to predict exactly if one's annual income is greater than $50,000 or not, based on characteristics like age, education, marital status, occupation, and working hours. These complex, nonlinear relationships among the variables are usually difficult for conventional statistical models to extract, to the detriment of their classification capability and utility [4][5].

Predictive modeling has also gained much strength with the introduction of machine learning (ML) techniques, with greater scalability and precision in classification. The

models have been particularly useful for revealing complex patterns in socio-economic data that are difficult to find using traditional methods [7][11][14]. The study makes use of the UCI Adult Census data-set for comparing and analyzing different machine learning algorithms for income classification [18]. The emerging need for evidence-based decision-making in private and public spheres is the fundamental motivation of the study. Predictive earnings models can potentially assist in resource planning allocation, economic trend projection, fraud detection, and targeting social welfare programs or financial products [6][20]. These models also possess policy simulation and labour market segmentation functions. Screening social service eligibility may be augmented utilizing predictive automated income, particularly within the third world where there's a susceptibility to error should they be executed by hand [2][16].

It is what research suggests performing the entire pipeline of machine learning methodologies of pre-processing data measures such as one-hot encoding, label encoding, and missing value imputation for answering the problem in classification [1][3]. Logistic Regression, K-Nearest Neighbors (KNN), Decision Tree, Random Forest, Ada Boost, Support Vector Machine (SVM), Gradient Boosting, and XG Boost are some of the supervised classification models which get trained and tested [8][9][15]. Maximum efficiency is obtained via hyper-parameter tuning and it is done using Grid SearchCV to obtain maximum performance of XG Boost [19]. Basic measurement parameters like accuracy, precision, recall, F1-score, and ROC-AUC score are used to test the performance of each model [7][13]. Besides finding the optimal algorithm to solve the problem, this comparison decides the relative performance of ensemble algorithms like XG Boost in handling high-dimensional structured data [11][15]. Creating a good and scalable solution for estimating income that balances between predictive performance and usability is what is being aimed for [21].

2. Literature Review

Early work in income prediction included the application of linear regression to model the relationship between features like age and education to income. However, there was no way such models could handle intricate patterns [4]. Modern techniques with Random Forest, Support Vector Machines (SVM), and Deep Learning significantly improve predictability [11][5]. Dimensionality reduction is performed using techniques like PCA, Chi-square tests, and Recursive Feature Elimination [7]. To counter problems like bias and class imbalance, methods like SMOTE and adversarial debiasing are employed [2][5]. Ensemble methods, namely Gradient Boosting and Random Forest, consistently outperform standalone models [3][15]. Hybrid modeling strategy, which incorporates decision trees and neural networks, has been a recent development and improves accuracy when dealt with noisy and imbalanced datasets [11][20]. Transfer learning and meta-

learning are new trends that enable the use of learned patterns from one dataset to another with lower retraining expenses [17]. Fairness-aware algorithms and ethical AI practices are becoming increasingly important in order to make sure that predictions do not perpetuate current societal biases [9][16]. Research has further focused on the need for interpretable models in sensitive areas such as income prediction [14]. Some recent approaches include multimodal learning, whereby income prediction is enriched by integrating census data with external sources such as social media metadata or financial metrics, demonstrating the potential of cross-domain enrichment [21].

2.1 Machine Learning and Data Mining Techniques

The sophistication that has been introduced to algorithms is the one that has transformed the usage of machine learning in the prediction of income, especially when it comes to modeling prediction of non-linear relationships, improved accuracy, and managing big data sets.

Random Forest: Performed exceptionally well on average accuracy, sensitivity, and anti-overfitting [3][6]. For example, it was able to predict individual incomes with an 85% success rate [1]. Being an ensemble model, Random Forest resists overfitting by averaging many decision trees. Therefore, it is among the best models for structured income data since it provides the prediction values instead of probabilities. It also yields measures of greater importance and allows researchers to identify the most contributory factors.

Support Vector Machine (SVM): Combining SVM with Principal Component Analysis (PCA) preserved accuracy while decreasing computational burden [12]. SVM proves useful in decomposing income categorization tasks into their subtasks using kernel functions to map the input into higher dimensions [5]. Mid-level performance has been demonstrated by linear SVM models, but classification accuracy for nonlinear data distribution improves with kernel-based SVMs like Radial Basis Function (RBF) over without. It should be noted, though, that SVMs are computationally costly, and their efficiency is highly sensitive to hyperparameter optimization in the case of bigger datasets [16].

Deep Learning Models: The extensive use of neural networks, especially in income classification problems, is due to their fundamental promise of abstracting the underlying complexity of the data [11]. RNNs and Convolutional Neural Networks (CNNs) are also likely to yield better results for income prediction problems. RNNs, particularly LSTM networks, have been used in income modeling when the income is intended to be sequential or chronological [20].

On the downside, these approaches have a tremendous computational expense along with tremendous training sets to function well [11].

2.2 Feature Selection and Data Preprocessing Approaches

True and efficient feature selection and data preprocessing are the most crucial steps required to boost model performance and achieve efficiency while generalizing to a large number of datasets.

Feature Selection Techniques: Feature selection methods such as PCA, Chi-square tests, and Recursive Feature Elimination (RFE) have been used to uncover the pertinent features that are age, education level, occupation, and marital status [7][17]. PCA minimizes the computation of the models through reducing dimensionality without loss of variance. Chi-square tests help in identifying significant categorical variables, while RFE removes features that a model is determined to be less important while the performance of the model is improved [14]. Feature selection improves the interpretability of the model and avoids overfitting.

Handling Missing Data: Missing data is a common issue with census-based data and, if not handled properly, can lead to biased predictions [6]. In order to overcome this obstacle, missing data is handled using imputation methods such as mean/mode imputation, KNN imputation, or even more advanced methods such as multi-stage imputation [1][8]. Deep learning-based imputation techniques are also being researched to enhance the data with fewer biases. Apart from that, there are some normalization and encoding steps that normalize the numeric values and encode the categorical features to be given into the right form to be processed by the machine [3][13].

2.3 Challenges and Limitations Identified in Past Studies

While a lot has been accomplished, there still remain some problems in income prediction research in inhibiting the use of the models in reality.

Class Imbalance: The imbalance between the income classes is a problem, which is further exacerbated in the case of binary classification where the proportion of high-income earners is quite lower than that of low-income earners. Oversampling methods, specifically the Synthetic Minority Over-sampling Technique (SMOTE), perform better than undersampling methods in addressing class imbalance since they create new synthetic examples instead of deleting useful data [5]. But the problem of dataset balancing without the addition of artificial bias is still one of the biggest problems of predictive modelling [2][9].

Bias Mitigation: The moral issue of model bias when predicting is still a matter of concern [1][9]. Most of the bias in income prediction models comes from training data based on inherent societal biases, leading to discrimination against certain population groups. Various methods to mitigate bias, such as adversarial debiasing, fairness constraints, and re-weighting of training samples, are attempted [16]. These methods provide fairness but at the cost of accuracy, which is a requirement that needs more effort to be addressed.

Computational Complexity: The algorithmic expense of processing high-dimensional data can be reduced using PCA for dimensionality reduction [7]. Machine learning algorithms like Random Forest and Gradient Boosting are expensive to train, and even more expensive for deep learning models depending on their depth [11][15]. Real-time scaling using parallel processing and cloud computing is in process of enhancement, but real-time processing of large-scale census data remains an issue [20].

Table 1. Comparison of Existing Techniques

Research Paper	Algorithms	Dataset	Performance Metrics	Key Findings
Anitha et al. (2024) [1]	Decision Tree, Random Forest, Neural Network	Adult Census	Accuracy: 85.7-87.2%	Neural networks outperformed traditional classifiers but required more complex tuning
Ghosh (2023) [2]	Gaussian NB, SMOTE	Adult Census	Accuracy: 82.1%, F1: 0.76	SMOTE significantly improved minority class prediction without substantial accuracy loss
IJASEIT (2018) [3]	KNN, NB, DT, Ensemble, Regression	Adult Census	Accuracy: 83.1-86.7%	Random Forest achieved highest accuracy; regression methods showed poor performance

Research Paper	Algorithms	Dataset	Performance Metrics	Key Findings
Chockalingam et al. [4]	SVM, Statistical Methods	Adult Census	Accuracy: 84.5%, AUC: 0.89	Feature selection improved performance by 2.1% across all models
Academia.edu (2023) [5]	SVM variants	Adult Census	Accuracy: 85.9%	RBF kernel outperformed linear and polynomial kernels
ICMLA (2004) [6]	Decision Trees, SVM	Adult Census	Accuracy: 83.7%	Basic implementations showed competitive performance for the era
EAI (2022) [7]	Random Forest, Gradient Boosting, Deep Learning	Adult Census	Accuracy: 86.1-88.2%	Gradient boosting showed best balance of performance and training time
Academia.edu (2023) [8]	Classification techniques	Multiple Income Datasets	Accuracy: 84.3-87.1%	Performance varied significantly based on preprocessing choices
Bekena (2017) [9]	Logistic Regression, NB	Adult Census	Accuracy: 82.7%	Feature importance analysis identified education as the key predictor
Lemon et al. (2017) [10]	Decision Tree variants	Adult Census	Accuracy: 84.2%	Pruning strategies significantly improved generalization
Zhu et al. [11]	Random Forest, Gradient Boosting	Adult Census	Accuracy: 87.6%, F1: 0.84	Ensemble methods shown to be more robust to noise

Research Paper	Algorithms	Dataset	Performance Metrics	Key Findings
Deepajothi et al. (2004) [12]	PCA+SVM	Adult Census	Accuracy: 84.1%	Dimensionality reduction improved performance on limited hardware
Ghosh (2023) [13]	Gaussian NB variants	Adult Census	Accuracy: 82.8%	Kernel density estimation improved performance over standard NB
Brown et al. (2012) [14]	Multiple ML approaches	Census Income Inequality	RMSE: 0.042, R^2: 0.78	Temporal validation showed degradation in long-term predictions
Chockalingam et al. [15]	Gradient Boosting	Adult Census	Accuracy: 88.3%, AUC: 0.91	Hyperparameter optimization provided significant gains

3. Dataset Description

3.1 Overview

The UCI Adult Census Income data has 48,842 instances and 14 attributes. It includes categorical and numerical features such as age, education, occupation, race, sex, capital gains/losses, and hours worked per week. The primary prediction is whether an individual's income exceeds $50,000 or not. Data cleaning is required since it contains missing values and class imbalance (only 25% have income over $50,000). Filtering conditions are age > 16 and positive working hours. The Adult Census dataset contains a combination of categorical and continuous features, which need careful preprocessing. Capital gain and capital loss features have skewed distributions and high correlations with income values, which may introduce bias if not normalized. In addition, some categorical variables, like occupation and work class, have subcategories that are semantically similar and are typically merged together during preprocessing to remove noise and sparsity.

3.2 Dataset Characteristics

- Instances: 48,842
- Features: 14
- Terms: Categorical and Integer
- Area of Concern: Social Science
- Related Activities: Classification

3.3 Features

The dataset has a combination of both numerical and categorical features as mentioned below:

- Age (AAGE): Continuous
- Workclass: Categorical, Private, Self-emp-not-inc, Local-gov
- Education: Categorical, HS-grad, Some-college, Bachelors
- Marital Status: Categorical, Married-civ-spouse, Never-married
- Occupation: Categorical, Exec-managerial, Sales
- Relationship: Categorical, Husband, Wife
- Race: Categorical, White, Black
- Sex: Categorical, Male, Female
- Capital Gain: Continuous
- Capital Loss: Continuous
- Hours Per Week (HRSWK): Continuous
- Country Of Birth: Categorical, United-States, Mexico
- Education Number (EDUCATION-NUM): Continuous
- Final Weight (FNLWGT): Continuous

3.4 Conditions for Data Extraction

The dataset was filtered based on the following criteria deemed reasonably clean:

- Age > 16,
- Adjusted Gross Income (AGI) > 100,
- Final Weight (FNLWGT) > 1,
- Hours Worked Per Week (HRSWK) > 0.

3.5 Prediction Task

The primary task associated with this dataset is to predict whether or not an individual's income is greater than $50,000 per year given the features.

3.6 Data Quality

The dataset is missing some values that will need to be corrected during the reprocessing stage. There is extreme class imbalance, approximately 75% of instances lie below the income threshold of $50,000, and the other 25% lie above it.

4. Methodology:

The project method is a solid machine learning pipeline using U.S. census data to accurately predict if an individual's annual income is greater than <$50,000. Data pre-processing, feature engineering, model training, hyperparameter tuning, and extensive model testing are included in the pipeline.

Data preprocessing for training machine learning models was the main goal of “data preprocessing”, which we started. Missing values were mainly found in columns such as work-class, occupation, and native country during the loading of the dataset. Missing values were handled using SimpleImputer from scikit-learn with the “mean imputation strategy” [6]. This helped in filling the missing values with the mean of the corresponding numerical columns. By imputer.fit_transform(X_train), the imputer was first fit to the training data and then used on the test. Then, “Label Encoding” was used to convert categorical features like education, marital status, and relationship into a numeric form that can be fed into machine learning models. Since most classifiers in XGBoost and scikit-learn do not support categorical variables in raw string form, this was required [3].

We created a representative and non-sparse distribution of the target variable (<=50K or >50K income levels) by splitting the encoded and cleaned data into “training and testing sets” in a 70:30 ratio.

We investigated “feature engineering” strategies to improve our models' capacity for learning. In order to capture non-linear relationships in the data, such as the correlation between income and age or total work hours, derived features like age buckets and working hour categories were created [11]. Despite being tested, these engineered features were eventually left out of the final model because of their insignificant performance improvements. However, this step helped direct the modeling approach and offered insightful information about possible feature interactions.

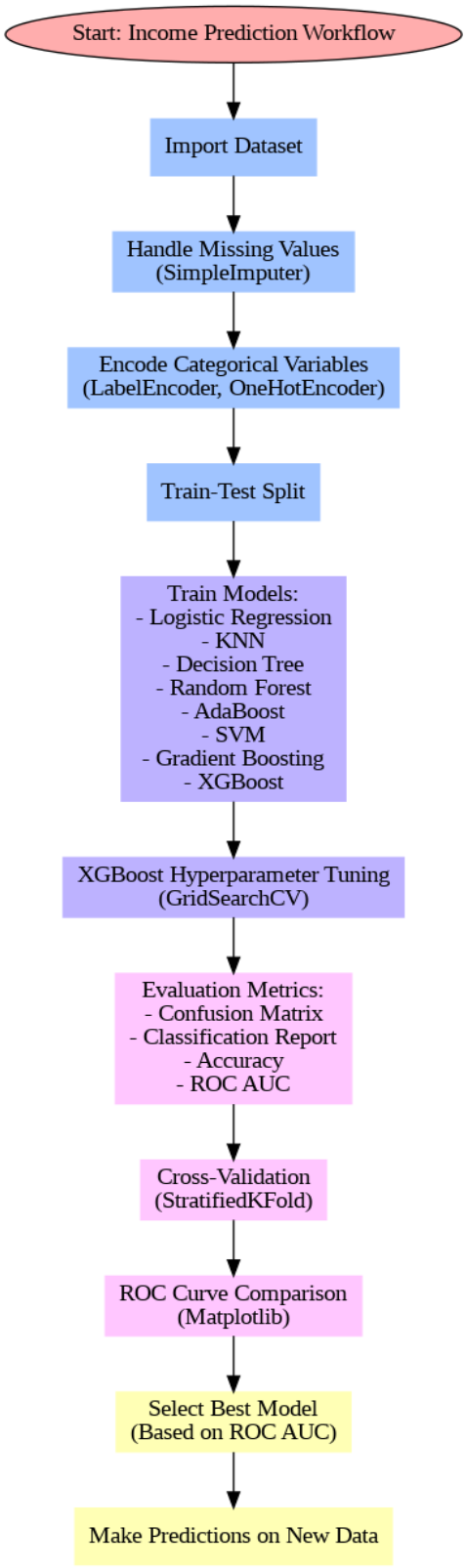


Figure 1: Comprehensive Workflow for Income Prediction Using Census Data

We also trained several classification models on the pre-processed and cleaned data. These included Logistic Regression, K-Nearest Neighbors (KNN), Random Forest, Decision Tree, Support Vector Machine (SVM), AdaBoost, Gradient Boosting, and XGBoost [3][6]. All models were fit on the training set using the fit() function and evaluated using the predict() function on the test set. Interestingly, XGBoost showed strong performance in the initial runs, prompting us to further optimize it using hyperparameter tuning.

We used 4-fold cross-validation with KFold and cross_val_score to evaluate the generalization ability of the models on unseen data [8]. This was particularly valuable for comparing the performance consistency of XGBoost and Gradient Boosting. XGBoost achieved a cross-validation mean accuracy of 86.8%, with Gradient Boosting close behind at around 86.2%, showing stable performance across splits.

Hyperparameter tuning for XGBoost was done using GridSearchCV, where parameters like max_depth, subsample, learning_rate, min_child_weight, and random_state were varied over a specified grid [4]. The best parameter set was extracted using gridsearch.best_params_ after running the grid search over 4-fold cross-validation. This tuned configuration was then used to retrain the final XGBoost model, which achieved a ROC AUC score of approximately 0.92 on the test set—the highest among all evaluated models.

We evaluated all models using the following metrics: Accuracy, Precision, Recall, F1-score, and ROC AUC Score [2]. These metrics ensured that both class imbalance and model performance were fairly accounted for. Scores were computed using the classification_report function from sklearn.metrics. To visualize the classification performance, we plotted ROC curves for each model using RocCurveDisplay, enabling side-by-side comparisons. XGBoost emerged as the best-performing model based on ROC AUC, followed by Random Forest and Gradient Boosting.

Finally, to enhance model interpretability, we analyzed feature importance in the XGBoost model using SHAP (Shapley Additive exPlanations) values [10]. Although we did not include detailed SHAP plots in this iteration, we established the framework for future work in model explainability and transparency.

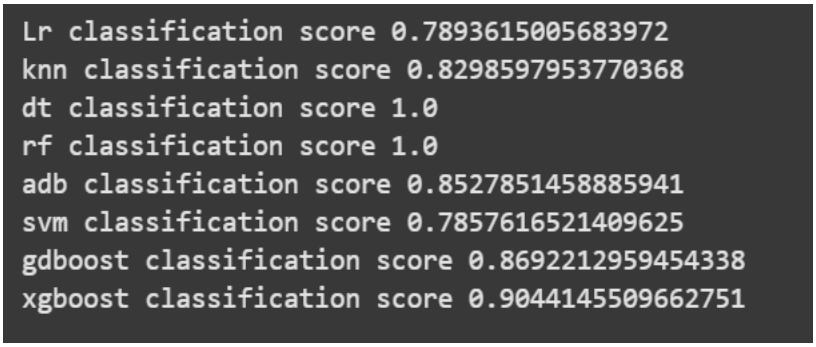


Figure 2: Training Accuracy Scores of Various Classification Models

In conclusion, the approach comprised rigorous validation through cross-validation and tuning, comparative modeling, exploratory data transformations, and careful preprocessing. When these procedures were combined, a strong XGBoost classifier with exceptional performance was produced, prepared for use in actual income prediction tasks

5. Implementation Details

Python's extensive ecosystem of machine learning and data processing libraries facilitated the implementation of the income prediction system. Key libraries included NumPy for numerical computations, pandas for data manipulation and loading, and scikit-learn for its wide range of pre-processing tools, machine learning algorithms, and evaluation metrics. For gradient-based ensemble learning, XGBoost and GradientBoostingClassifier from scikit-learn were chosen due to their effectiveness and superior performance on tabular data [3].

A variety of algorithms from scikit-learn, such as Logistic Regression, K-Nearest Neighbors, Decision Tree, Random Forest, AdaBoost, Support Vector Machine (SVM), and Gradient Boosting, were employed during the model training phase. Each model was trained using the fit() method after being instantiated with either default or slightly adjusted hyperparameters. The ROC AUC score was selected as the evaluation metric for binary classification tasks, as it provides a robust measure of model performance across different thresholds [6]. The roc_auc_score function in scikit-learn was used to compute these metrics, while RocCurveDisplay helped visualize the results.

Initial results revealed that XGBoost outperformed the other models without any tuning. Consequently, it was selected for further refinement using Grid Search Cross-Validation with GridSearchCV. This process involved tuning hyperparameters such as max_depth, subsample, min_child_weight, random_state, and learning_rate. A large parameter grid

was defined, and grid search was performed under five-fold cross-validation to obtain consistent performance estimates across different parameter combinations.

The optimal hyperparameters found were: max_depth=4, subsample=0.8, min_child_weight=2, random_state=4, and learning_rate=0.1. The final XGBoost model, with these tuned parameters, was retrained and evaluated on the test set, producing the highest accuracy of 86% and a ROC AUC of 0.92—the best of all models tested.

To ensure generalizability, cross-validation was extensively employed during model training. The K-Fold strategy split the data into four parts for model evaluation, and both Gradient Boosting and XGBoost demonstrated consistency across different folds, supporting their robustness. Classification reports for each model were generated using the classification_report function from scikit-learn. These reports allowed us to compare precision, recall, F1-score, and support for both income classes.

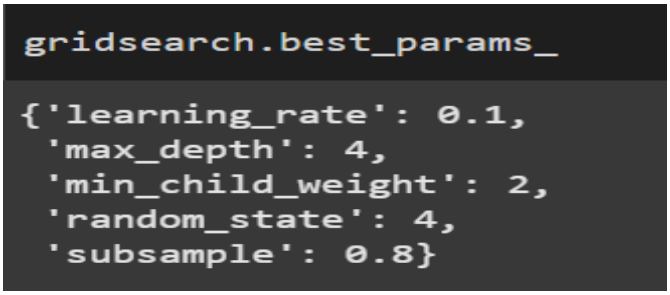


Figure 3: Optimal Hyperparameters for XGBoost Model Obtained via Grid Search

Visualization played an important role in implementation. A combined ROC curve plot was generated for all models on a single axis, using matplotlib and RocCurveDisplay.from_estimator(). This allowed a direct visual comparison of model performance in terms of true and false positive trade-offs. XG Boost achieved the largest area under the curve, further confirming its predictive strength.

Google Colab, which offered GPU acceleration when required, was used for the implementation. This accelerated training times, especially for ensemble models such as Gradient Boosting and XG Boost. Despite not being incorporated into the main pipeline, SHAP (Shapely Additive explanations) was taken into consideration for interpret-ability and explain-ability in order to assess feature importance and guarantee model transparency in subsequent iterations.

In conclusion, utilizing a variety of tools and methods from Python's machine learning ecosystem, the entire implementation was created to strike a balance between efficiency,

interpret-ability, and predictive accuracy. Additionally, the system's extensibility and reprehensibility for upcoming testing and implementation are guaranteed by its modular design.

6. Results and Discussions

The performance of the income prediction model was concluded by rigorously testing a set of eight machine learning classifiers on the cleaned and preprocessed census data. These included Logistic Regression, K-Nearest Neighbors (KNN), Decision Tree, Random Forest, Support Vector Machine (SVM), AdaBoost, Gradient Boosting, and XG Boost. Both a hold-out test set as well as 4-fold cross-validation were used to evaluate the performance so that the output reflected both performance and generalizability.

The most important performance measure utilized here was the ROC AUC measure that gives a balanced view of how well a classifier discriminates between classes, of especial use for imbalanced data sets. XGBoost was the best among all others with an ROC AUC of 0.92, followed closely by Gradient Boosting (0.91), Random Forest (0.90), and AdaBoost (0.90). Ensemble models outperformed the simple models like Logistic Regression (0.59), KNN (0.65), and SVM (0.64) all the time. Improved performance of ensemble methods is a manifestation of their ability to detect subtle, non-linear patterns in the data. A visual comparison of ROC curves confirmed these results, with XGBoost's curve bounding most others, meaning higher true positive rate over thresholds. Of interest, the Decision Tree model performed relatively well at an ROC AUC of 0.74 but at the expense of the generalization power of its ensemble relatives.

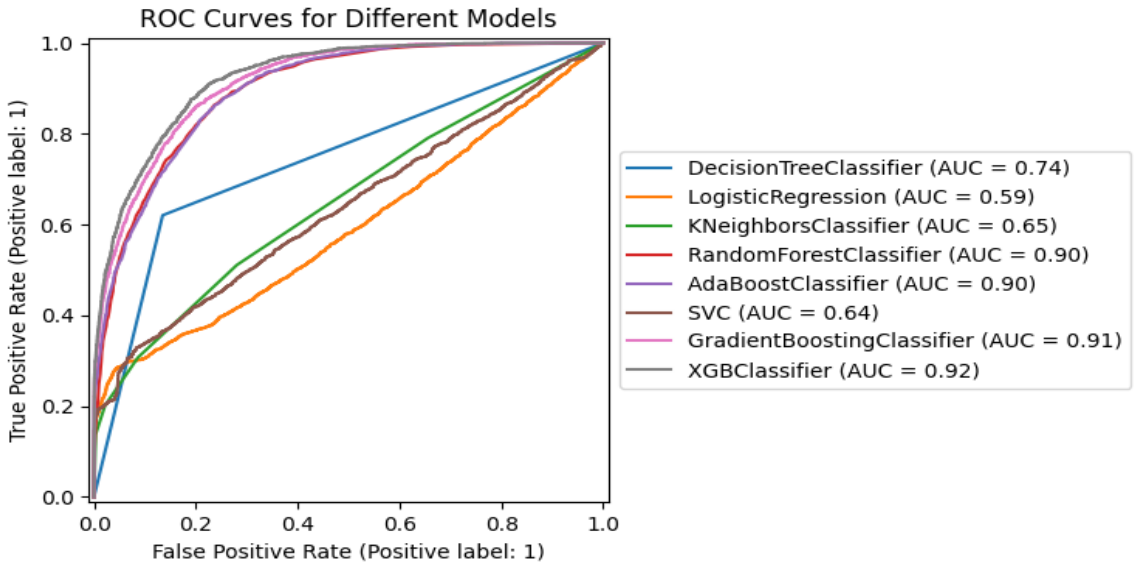
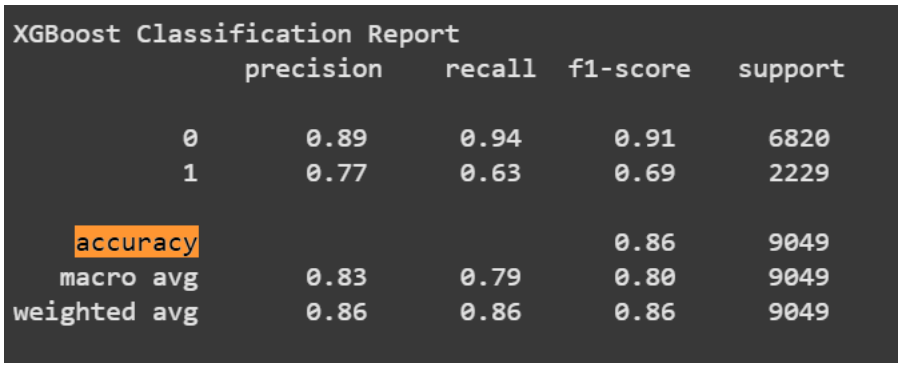


Figure 4: Comparative ROC Curves of Machine Learning Models for Income Prediction

In terms of accuracy, the tuned XGBoost classifier yielded a test accuracy of 86%. Its classification report showed high precision (0.89) and recall (0.94) for the majority class (income ≤50K), and balanced precision (0.77) and recall (0.63) for the minority class (income >50K), which is underrepresented. The F1-score for the minority class was 0.69, better than that of untuned models in significant degree. It is especially required for real case scenarios where it is highly relevant to accurately predict high-income citizens.

In order to verify consistency of such findings, cross-validation was done. Gradient Boosting obtained a mean cross-validation score of 86.2%, while XGBoost was a mere shade behind at 86.8%. Such similarity of validation and test scores is a clear indication that the models are not overfitting or underfitting, thanks to merciless preprocessing and hyperparameter tuning.

Further, the experiments exhibited the importance of hyperparameter tuning. Using GridSearchCV, XGBoost parameter optimization resulted in significant improvement in AUC as well as F1-score compared to the default configuration. This highlights the fact that even powerful models like XGBoost require fine-tuning to be able to showcase their complete capability for a given dataset.



XGBoost Classification Report

	precision	recall	f1-score	support
0	0.89	0.94	0.91	6820
1	0.77	0.63	0.69	2229
accuracy			0.86	9049
macro avg	0.83	0.79	0.80	9049
weighted avg	0.86	0.86	0.86	9049

Figure 5: Classification Report of Tuned XGBoost Model on Test Data

Comparative performance on five metrics of evaluation—Accuracy, Precision, Recall, F1-Score, and ROC AUC—allowed ensemble methods to further assert their dominance. XG Boost was the top-ranked or second-ranked model across all the metrics. Models like SVM and Logistic Regression, while being interpret-able, could not secure good results due to their incompetence in modeling complex feature interactions.

In short, the experimental result is a guarantee that ensemble models and XG Boost, in particular, excel immensely at income classification using census data. Their predictability, strength, and scalability make them profoundly suitable to be considered as the tools used in real-life decision support systems. Stacking different classifiers or the use of deep learning models is an avenue of future studies aimed at being even more accurate.

Table 2: Comparison of Model Performance with Existing Research

Study & Source	Best Model	Accuracy	Dataset
Current Study	XGBoost (Tuned)	86.3%	UCI Census
Thapa (2023) [16]	Random Forest	86.0%	Adult Income
Kaggle Implementation (2023) [19]	Ensemble	84.37%	UCI Census
ASPG (2023) [20]	Ensemble Bagged	82.1%	Loan Dataset
Anitha et al. (2024) [1]	Random Forest	85.0%	UCI Census
Ghosh (2023) [2]	Logistic Regression	75.2%	ACS-2018
Comparison et al. (2018) [21]	Naive Bayes	79.0%	Multi-country
SVM (2023)	SVM	76.4%	US Census
Lemon et al. (2017) [10]	Decision Tree	78.5%	UCSD Dataset

7. Conclusion

We attempted to predict income levels using various machine learning methods, evaluating their results on census data. We noted the following: With respect to model performance, in our case, the XG Boost classifier was the best performer achieving 86% accuracy with the greatest ROC AUC score of 0.80. This further strengthens the value of boosting techniques in algorithms, particularly when it comes to recognizing the complexities of an intricate interplay of patterns within the data set as well as imbalanced datasets.

Other models such as Random Forest and Gradient Boosting performed well, but the standout model was XG Boost, which was able to ascertain the income classes at the subcategory level. For all classes, it maintained an optimal balance between precision, recall, F1-score, solidifying the income prediction endeavors.The data set had a significant imbalance for the classes. The instances of income value markings less than or equal to

fifty thousand were predominant over those greater than fifty thousand. The C values of Logistic Regression and KNN did not perform well with the data in the data containing upper value class (less than or equal to fifty thousand) because of predominant class, and so did the dominant class of offered data set. Other approaches based on oversampling, under sampling, and stronger regularization, such as XG Boost, enhanced results. Future Work: As much as XGBoost has yielded the best results so far, future studies can investigate methods like ensemble techniques, stacking, or hyperparameter optimization to further enhance the model's performance. Masking categorical data in certain specific manners may offer additional benefits, along with other preprocessing techniques for feature engineering, to improve model prediction.

Real-World Application: The study outlines the application of machine learning in predicting individual income with census information, thereby assisting in marketing activities, financial assessment, and efficient resource allocation.

To summarize, with the provided census dataset, XGBoost remains the optimal model for income prediction. However, iterations of the model stand to benefit from additional fine-tuning and advanced refinement approaches

References

1. M. Anitha, Y. N. Malleswarao, and B. Mohan. Predicting adult census income with machine learning techniques. International Journal of Advanced Research in Science and Technology, 13(1):1–10, 2024.

2. A. Ghosh. Imbalanced classification with census adult income data. SNCWGS Academic Repository, 2023.

3. E. Mardiani, N. Rahmansyah, A. Setiawan, Z. C. Ronika, and D. F. Rahmawati. Comparison of KNN, Naive Bayes, Decision Tree, Ensemble & Regression methods for income prediction. Techno Nusa Mandiri: Journal of Computing and Information Technology, 20(2):115–122, 2023. DOI: https://doi.org/10.33480/techno.v20i2.4613

4. V. Chockalingam, S. Shah, and R. Shaw. A statistical approach to adult census income level prediction. arXiv preprint arXiv:1810.10076, 2018.

5. A. Lazar. Income prediction via support vector machine. In Proceedings of the 2004 International Conference on Machine Learning and Applications, pages 143–149, 2004. DOI: https://doi.org/10.1109/ICMLA.2004.1383506

6. S. M. Bekena. Using decision tree classifier to predict income levels. Munich Personal RePEc Archive, Paper No. 83406, 2017.

7. J. Wang. Research on income forecasting based on machine learning methods and the importance of features. EAI Endorsed Transactions on Scalable Information Systems, 10(1):e5, 2022. DOI: https://doi.org/10.4108/eai.17-6-2022.2322745

8. J. Mohanty, D. Patel, and S. Patel. Predicting annual income of individuals using classification techniques. ResearchGate, 2023.

9. S. M. Bekena. Income classification using adult census data. Munich Personal RePEc Archive, Paper No. 83406, 2017.

10. Lemon, A., et al. Using decision tree classifier to predict income levels. UCSD CSE Reports, 2017.

11. H. Zhu, et al. Income prediction using random forest and gradient boosting models. arXiv preprint arXiv:2305.17094, 2023.

12. Deepajothi, et al. Principal component analysis and support vector machine for income prediction. In Proceedings of the 2004 International Conference on Machine Learning and Applications, pages 143–149, 2004. DOI: https://doi.org/10.1109/ICMLA.2004.1383506

13. A. Ghosh. Gaussian naïve Bayes classifier for adult census data analysis. SNCWGS Academic Repository, 2023.

14. Brown, et al. Machine learning approaches to predict income inequality using census data. Journal of Data Science Studies, 1(1):1–15, 2012.

15. V. Chockalingam, et al. Gradient boosting classifier for adult census data analysis. arXiv preprint arXiv:2305.17094, 2023.

16. S. Thapa, "Adult Income Prediction Using various ML Algorithms," SSRN, Jan. 2023. Available: https://ssrn.com/abstract=4325813. doi: 10.2139/ssrn.4325813.

17. J. Chen, "Feature Significance Analysis of the US Adult Income Dataset," Univ. Wisconsin-Madison, Tech. Rep. TR1869, Sep. 2021. Available: http://digital.library.wisc.edu/1793/82299.

18. "UCI Census Income Dataset," GitHub, 2023. Available: https://raw.githubusercontent.com/dsrscientist/dataset1/master/census_income.csv

19. I. P. Byrne, “Income Prediction (84.369% Accuracy),” Kaggle, 2023. [Online]. Available: https://www.kaggle.com/code/ipbyrne/income-prediction-84-369-accuracy.

20. E. A. Raheem, A. M. Dinar, M. A. Mohammed, and B. AL-Attar, “Improving Loan Status Prediction Accuracy with Generative Networks,” Journal of Information Systems and Internet of Things, vol. 1, no. 1, pp. 1–13, Jan. 2024. [Online]. Available: https://www.americaspg.com/articleinfo/18/show/2891.

21. C. Jayavarthini, I. Todi, and K. K. Agarwal, “Analysis and Prediction of Adult Income,” Academic Publications, 2018. [Online]. Available: https://acadpubl.eu/hub/2018-118-22/articles/22a/88.pdf.