Author: Shreeda Ghatpande

Couse: CS 6350

Github link: <https://github.com/Dashree/income_predictor.git>

**1. Task Introduction**

The goal of this project is to predict whether a person earns more than $50K per year, based on demographic and employment-related features. This goal is important as to help organizations to identify which demographic to help and focus to increase the earning capacity to help people. We will employing supervised learning strategies to make predictions as accurate as possible using deterministic techniques.

The task is to predict whether a person earns more than $50,000 annually, given certain features like age, education level, occupation, hours worked per week, etc. This is a **binary classification** problem, where the target variable is income (greater than $50K or not).

**2. Code Description**

Data: folder contains the train and test data provided

Src : Containd all the necessary source code

Report : Contains this report

Images: Dataset visualization images

**3. Dataset Exploration**

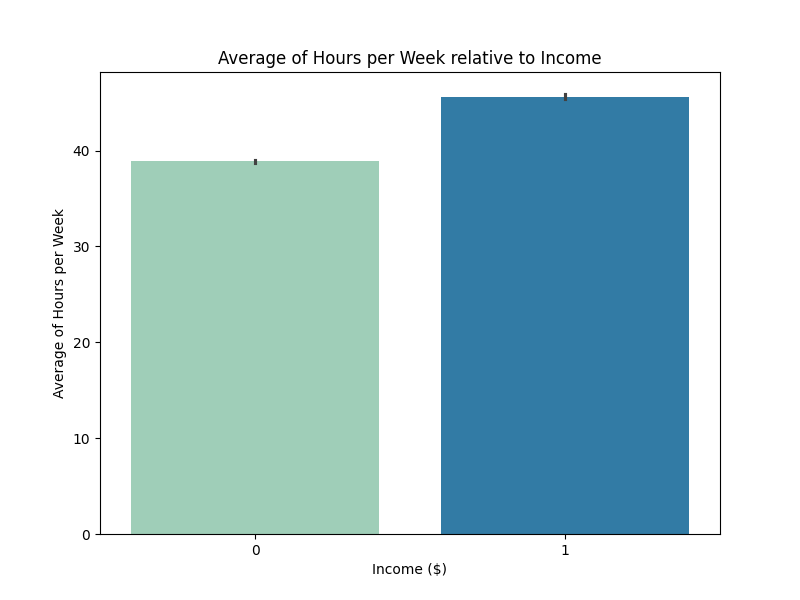
The dataset provided with fifteen features that can used to determine the income of individuals namely, age, workclass, fnlwgt, education, education number, marital status, occupation, relationship, race, sex, capital gain, capital loss, hours per week, native country.

The dataset typically includes the following features:

* **age**: The age of the individual.
* **workclass**: The type of employment (e.g., private, self-employed).
* **fnlwgt**: The final weight assigned to the instance (often used to reflect the sampling weight).
* **education**: The highest level of education attained (e.g., Bachelors, Masters).
* **education\_num**: The numeric representation of education level.
* **marital\_status**: Marital status of the individual.
* **occupation**: Occupation of the individual (e.g., tech-support, sales).
* **relationship**: The relationship status (e.g., husband, wife).
* **race**: The race of the individual.
* **sex**: Gender of the individual.
* **capital\_gain**: Amount of capital gain.
* **capital\_loss**: Amount of capital loss.
* **hours\_per\_week**: Hours worked per week.
* **native\_country**: The country of origin.
* **income**: The target variable indicating whether income is greater than $50K.

On first observation we can see that education and education number are duplicate one can removed without loss of any information. There are 25000 such rows data available for training. With 23843 data points for testing.

A graph of a number of people

Description automatically generated

**4. Data Preprocessing**

Data preprocessing is a critical step in the machine learning pipeline. The preprocessing steps include:

* **Handling missing values**: The dataset may contain missing values, which need to be handled (e.g., imputation or removal).
* **Encoding categorical features**: Many features like workclass, education, occupation, etc., are categorical and need to be encoded numerically for machine learning algorithms. This can be done using techniques like One-Hot Encoding or Label Encoding.
* **Feature scaling**: Machine learning algorithms perform better when features are scaled. Standardization or Min-Max scaling is used for this purpose.
* **Feature engineering**: Additional features may be created or transformed to improve model performance. For example, combining features like capital\_gain and capital\_loss might create a feature representing net capital.  
  **A graph of blue bars

  Description automatically generated with medium confidence**
* **Removing duplicate entries:** Removing some duplicate values present in train dataset. In this dataset there are 14 such rows.

**5. Machine Learning Models**

Several machine learning algorithms can be used to predict income. Some of the most common techniques for classification problems are:

* **AdaBoost**: A simple, interpretable model for binary classification that predicts the probability of an outcome.
* **Decision Trees**: A non-linear model that splits the data at various decision points.
* **Random Forest**: An ensemble method that combines multiple decision trees to improve accuracy and reduce overfitting.
* **Gaussian Naive Bayes:** A powerful classification algorithm based on Bayes.

**6. Model Evaluation**

After training the models, the performance should be evaluated using suitable metrics:

* **Accuracy**: The proportion of correct predictions (not always ideal in imbalanced datasets).
* **F1-Score**: The harmonic mean of precision and recall.
* **ROC-AUC**: The area under the Receiver Operating Characteristic curve, which evaluates the trade-off between true positive rate and false positive rate.

In this case, since the dataset is likely imbalanced (i.e., fewer people earning more than $50K), metrics like **F1-score** will be more informative.

**7. Results and Analysis**

A typical workflow would involve:

1. Splitting the dataset into training and test sets (e.g., 80% for training, 20% for testing).
2. Training different machine learning models (e.g., Logistic Regression, Random Forest, SVM) on the training data.
3. Evaluating the models using the test set and comparing their performance.
4. Choosing the best-performing model based on evaluation metrics.

An example result might be:

* **GaussianNB**: Accuracy = 80.3%, ROC AUC = 85.5%, F1-Score = 66.7%.
* **Random Forest**: Accuracy = 85.9%, ROC AUC = 90.6% F1-Score = 78.5%.
* **AdaBoost**: Accuracy = 83.4%, ROC AUC = 88.05%, F1-Score = 80.05%.

**8. Conclusion**

Among the models tested, **Random Forest** yielded the best results, especially in handling the complexity and non-linear relationships present in the dataset. Feature engineering and careful preprocessing (like handling categorical features and scaling) significantly affect the model's performance.

Further improvements can be made by tuning hyperparameters using grid search or random search, incorporating more advanced algorithms, and ensuring the model is generalized well with techniques like cross-validation.

**9. Future Work**

* **Feature selection**: Investigating which features contribute most to the model’s predictions.
* **Deep learning**: Trying deep learning models like neural networks might offer improved performance on large datasets.
* **Class imbalance handling**: Implementing strategies like oversampling or undersampling for better handling of imbalanced data.