Artificial Intelligence in Adult Census Income Problem

Chao ZHU
Department of Computer Science and Engineering
Southern University of Science and Technology
Shenzhen, China
12210305@mail.sustech.edu.cn

***Abstract*——In this study, we explored several machine learning algorithms to predict the adult census income based on various demographic and socioeconomic features. Through the utilization of classification models such as logistic regression, random forest, and gradiant boosting, we aim to develop accurate predictive models for determining whether an individual earns more than $50,000 annually. In this report, we also compraed the performance of these models on this question.**

***Index Terms*—artificial intelligence, machine learning, statistic**

I. Introduction

THE analysis and prediction of adult census income play a pivotal role in understanding socio-economic dynamics and formulating targeted policy interventions. With the advent of advanced machine learning algorithms and the availability of extensive demographic data, researchers and policymakers have gained new tools to uncover patterns and trends in income distribution within a population.

This report aims to delve into the realm of predicting adult census income using machine learning techniques applied to an Adult Census Income dataset. By leveraging this dataset, which contains detailed information about individuals' demographics, education, occupation, and other relevant features, we seek to develop models that can accurately predict whether an individual earns more than $50,000 annually, thereby providing valuable insights into the factors influencing income levels.

The dataset was extracted from the 1994 Census bureau database by Ronny Kohavi and Barry Becker (Data Mining and Visualization, Silicon Graphics). The prediction task is to determine whether a person makes over $50K a year.

II. Preliminary

The specific data we care in the data set was as follows:

- age:the working age of each data sample, which is a numerical variable;
- workclass: type of work, where there are private, local government, etc., is a character type variable;
- education_num: the level of education of each sample;
- marital_status: marital status of each sample;
- occupation: the occupation of each sample;
- relationship: the family relationship of each sample;
- race: the race of each sample;
- gender: the sex of each sample;
- capital_gain: a capital gain is a profit that results from a disposition of a capital asset, such as stock, bond or real estate, where the amount realised on the disposition exceeds the purchase price;
- capital_loss: capital loss is the difference between a lower selling price and a higher purchase price, resulting in a financial loss for each sample ;
- hours_per_week: sample weekly working hours;
- native_country: the country where the sample is from;

We will use these attributes to train the model and evaluate the model by roc-auc-score, f1-score and accuracy.

The fomula for f1-score is:

$$\text{F1-score} = 2 \cdot \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

The fomula for accuracy is:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

among which

- TP = true positive, predict that the answer belongs to this category and is correct
- TN = true negative, predict that the answer belongs to the other category and is correct
- FP = false positive, predict that the answer belongs to this category and is not correct
- FN = false negative, predict that the answer belongs to the other category and is not correct

III. Methodology

The methodology mainly consist of the following two parts

1) Data visualization. In order to get a more intuitive view of the data and its relationship between the attributes and the result, we visualize the data to avoid overfitting or underfitting.
2) Data processing. We split the data into training set and testing set, and then we use the training set to train the machine learning models, and then use the testing set to evaluate the model.

A. Data Visualization

1) Overall infomation: The dataset provide 14 attributes related to the income and the porpotion of people that earn more than $50,000 annually is about 24.1%.

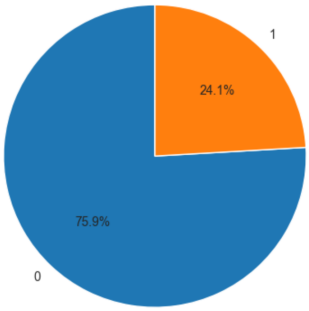

Fig. 1. overall situation of the dataset

2) Age distribution: We can see that those who earn more than $50,000 annually are usually between 25 and 65 years old.

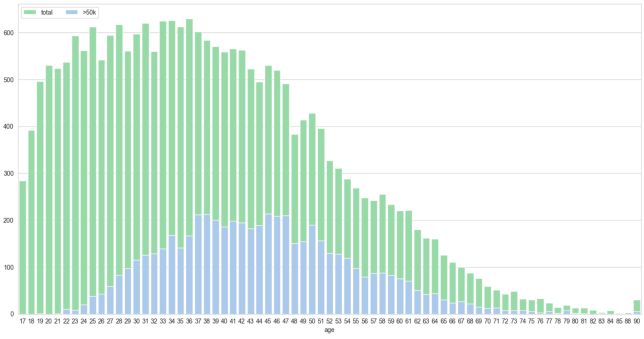

Fig. 2. age distribution of the dataset

3) Education distribution: We can see that those with higher salary usually receives better education. Those who graduates from professional school or has a doctoral degree have a chance of up to 70% to earn $50,000 annually.

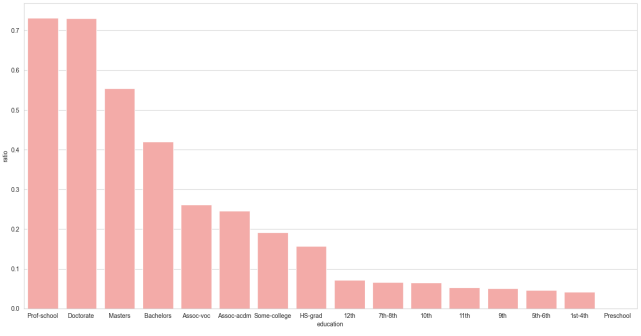

Fig. 3. education distribution of the dataset

4) Other factors: Other factors include family relationship, race, occupation, marriage, gender, home country, capital gain and loss are also very important to decide one's income, and the data in the dataset are as follows:

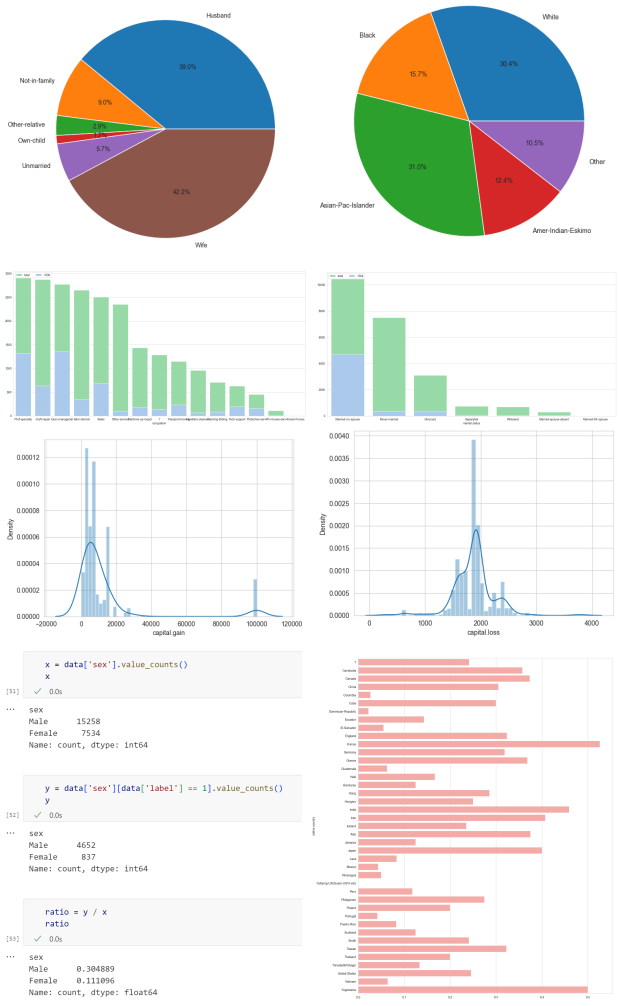

Fig. 4. other factors that influence income

B. Data Processing

In this part, we have got 6 algorithms, which are simple decision, decision tree, logistic regression, random forest, stochastic gradient descent and gradient boosting.

1) Simple decision: The simple decision will simply predict the one with the highest possiblity. In this problem, it will simply predict 0, or not able to earn 50k dollars annually.

2) Decision tree: A decision tree is a popular supervised machine learning algorithm used for both classification and regression tasks. It works by recursively partitioning the data into subsets based on the values of input features. At each node of the tree, a decision is made based on the feature that best splits the data into homogeneous subsets, aiming to maximize the purity or information gain. This process continues recursively until a stopping criterion is met, such as reaching a maximum tree depth or having data subsets with homogeneous target values.

3) Logistic regression: Logistic regression is a popular statistical method used for binary classification tasks. It's a type of regression analysis where the dependent variable is categorical, typically representing two classes, and the output is a probability value that indicates the likelihood of a given input belonging to one of the classes.

In logistic regression, the output of the model is transformed using the logistic function, also known as the sigmoid function, which maps any real-valued number to the range between 0 and 1. This transformation allows us to interpret the output as a probability.

The algorithm works by fitting a logistic curve to the data, which separates the two classes based on the input features. It optimizes the parameters of the logistic function to minimize the error between the predicted probabilities and the actual class labels. Here is a pseudo code of logistic regression:

Algorithm 1 Logistic Regression

function LOGISTICREGRESSION(X, Y, learning_rate, max_iterations)
  Initialize weights W and bias b randomly
  for $i = 1$ to $max\_iterations$ **do**
    $Z = XW + b$
    $A = \sigma(Z)$
    $l = -\frac{\sum_{i=1}^{m}(y^{(i)}\log(a^i)}{m} + (1 - y^i)\log(1 - a^i))$
    $dW = \frac{1}{m}X^T(A - Y)$
    $db = \frac{1}{m}\sum_{i=1}^{m}(A - Y)$
    $W = W - learning\_rate \times dW$
    $b = b - learning\_rate \times db$
  end for
  return W, b
end function

4) Stochastic gradient descent: Stochastic Gradient Descent (SGD) is an optimization algorithm commonly used in machine learning and deep learning for training models. Unlike traditional gradient descent, which computes the gradient of the cost function using the entire training dataset, SGD updates the model parameters incrementally for each training example. Here is a pseudo code of SGD:

Algorithm 2 Stochastic Gradient Descent

function STOCHASTICGRADIENTDESCENT(X, Y, learning_rate, epochs)
  Initialize model parameters randomly: θ
  for $epoch = 1$ to $epochs$ **do**
    for $i = 1$ to m **do**
      Sample a random training example: $(x^{(i)}, y^{(i)})$
      $\nabla_\theta J(\theta; x^{(i)}, y^{(i)})$
      $\theta \leftarrow \theta -$ learning_rate $\times \nabla_\theta J(\theta; x^{(i)}, y^{(i)})$
    end for
  end for
  return trained parameters: θ
end function

5) Random forest: Random Forest is an ensemble learning method for classification, regression, and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. It combines the predictions of multiple decision trees to improve generalizability and robustness over a single estimator. Here is a pseudo code of random forest:

Algorithm 3 Random Forest

function RANDOMFOREST(X, Y, num_trees, max_depth, num_features_per_splitS)
  Initialize an empty list of decision trees: $forest$
  for $i = 1$ to num_trees **do**
    Sample a subset of features of size $num\_features\_per\_split$ without replacement
    Sample a subset of data points from the training set with replacement
    Train a decision tree on the sampled data using the sampled features, with maximum depth max_depth
    Add the trained tree to the forest
  end for
  return $forest$
end function

6) Gradient boosting: Gradient Boosting is a powerful machine learning technique used for both regression and classification tasks. It builds a predictive model in the form of an ensemble of weak learners, typically decision trees, by sequentially adding them to minimize the loss function. Here is a pseudo code of gradient boosting:

Algorithm 4 Gradient Boosting

function GRADIENTBOOSTING(X, Y, number_of_trees, learning_rate)
  $F_0(x) =$ mean(Y)
  for $t = 1$ to T **do**
    $r_i = y_i - F_{t-1}(x_i)$ for $i = 1$ to N
    $h_t(x) =$ WeakLearner, i.e. decision tree(X, r)
    $F_t(x) = F_{t-1}(x) +$ learning_rate $\times h_t(x)$
  end for
  return $\{h_1, h_2, ..., h_T\}$
end function

IV. EXPERIMENTS

A. Environments

- machine: windows11 version 23H2, AMD R7 6800H
- python: 3.9.7
- IDE: VSCode
- dataset, source code and requirements are available at github

B. Dataset

The datasets used to test the algorithm was extracted from the 1994 Census bureau database by Ronny Kohavi

and Barry Becker (Data Mining and Visualization, Silicon Graphics). The prediction task is to determine whether a person makes over $50K a year. We use `traindata.csv` and `trainlabel.txt` to train and evaluate the model. We use 70% of the data to train the model and 30% of the data to evaluate the models' performance.

C. Results

The performance of the models listed in the table:

Model	f1-0	f1-1	roc-auc	accuracy
Simple	0.86	0.00	0.5000	0.76
Decision tree	0.88	0.61	0.7554	0.81
Logistic regression	0.91	0.67	0.9068	0.85
Stochastic gradient descent	0.91	0.66	0.9056	0.85
Random forest	0.90	0.66	0.8913	0.85
Gradient boosting	0.92	0.70	0.9218	0.87

TABLE I
PERFORMANCE OF THE MODELS

Here is the ROC curves and precision curves of the models:

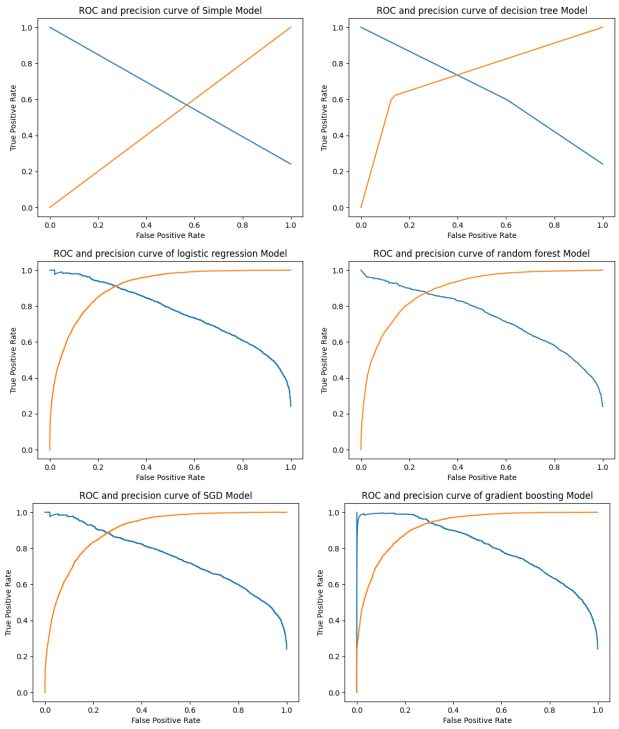

Fig. 5. ROC and precision curves of the models

D. Analysis

Based on the experiment results of Adult Income Census Prediction, the gradient boosting algorithm have almost the best performance. But this also have some limitations, and there are many factor that I did not take into consideration due to the limited time such as the hyper-parameter, thus the evaluation of the model is not that accurate.

V. Conclusion

In the course of this project, I have gained increased proficiency in using machine learning methods to solve a problem.

This experience has provided valuable insights and serves as a catalyst for further exploration and refinement in the realm of such problems.