Project Report: Adult Census Income

Chen Chang
Computer Science and Technology Department (of SUSTech)
Shen Zhen, China
12212739

***Abstract*—This document is the report for the third AI project, to create a classification model predicting incomes from adults based on the adult census dataset.**

***Index Terms*—Problem specification, preprocess data, logistical regression, svm, random forest, KNN, AdaBoost, K-fold cross validation.**

I. Problem Specification

Based on a small partition of data extracted from the 1994 Census bureau database by Ronny Kohavi and Barry Becker (Data Mining and Visualization, Silicon Graphics), I constructed several simple clarification models to predict whether a person can make over $50K a year. The features of a person includes:

- age: numerical working age.
- workclass: type of work, character type.
- fnlwgt: the number of observational representatives of in the corresponding state.
- education: the level of education.
- education.num: the schooling year.
- marital.status: marital status.
- occupation
- relationship: the family relationship, husband or wife or not in family, etc.
- race
- sex
- capital.gain: profit that results from a disposition of a capital asset.
- capital.loss: difference between a lower selling price and a higher purchase price.
- hours.per.week: weekly working hours.
- native.country: the country where the sample is from.
- income (trainlabel.txt): income label for which 1 means greater than 50K and 0 means less than or equal to 50K.

Now we have two training files, traindata.csv and trainlabel.txt which stores the input features and output target of an AI model. There is another testdata.csv which stores the test data and we need to predict the label and store them into testlabel.txt.

II. Data Preprocess

A. Exploratory Data Analysis

First after fetching the training data, we need to analyze the data so we can have a basic understanding and select possible best model structure based on the knowledge.

B. Feature Scaling

I chose to scale all numerical attributes of training data so that they have a mean of 0 and a standard deviation of 1 before they are fed to the machine learning model. This motion can speed up the converging of machine learning models.

C. Categorical Attributes to one-hot Vectors

I used one-hot encoding to convert categorical variables into binary columns. Each category becomes a column with 1 for the presence of that category and 0 for others.

D. Imputation for Missing Data

Some data in train data is undefined which is denoted by ”?” in csv file. In order to train a model, practically we need to replace those undefined fields into something that can join in the model calculation as a number. I chose to replace a numerical missing number with the median of that column, and replace a categorical missing value with the most frequently appeared one of that column.

III. Model Selection

Logistic Regression is simple to implement and is suitable for binary classification. It has a high interpretability. Also it can handle large feature spaces with linear increasing comptational cost. Its weakness is that it uses linear formula w.r.t attributes to apply into sigmoid function, which is limited when encountered nonlinear problems.

Decision Trees are suitable for non-linear relationships, can generate the importance of features, and is powerful to generate nonlinear decision boundaries. Its weakness is that it is easy to prone to overfitting and we have to prune it to avoid overfitting. But pruning parameters are hard to choose and can make the model quite unstable. To overcome these drawbacks we can use multiple trees and let them vote for the final decision. Based on the process of generating trees we have two algorithms, Random Forest and AdaBoost. These models are less sensitive to overfitting and are more stable.

K Nearest Neighbors(KNN) can capture local patterns, and it is specially non-parametric. Its weakness is that it is sensitive to irrelevant features and it is very computationally expensive. Also it has a really low interpretability.

Support Vector machines(SVM) is effective in high-dimensional spaces, its weakness is that it needs many parameter pruning so getting its best configuration is quite difficult. It also has a low interpretability.

IV. Model Evaluation

A. K-fold Cross Validation

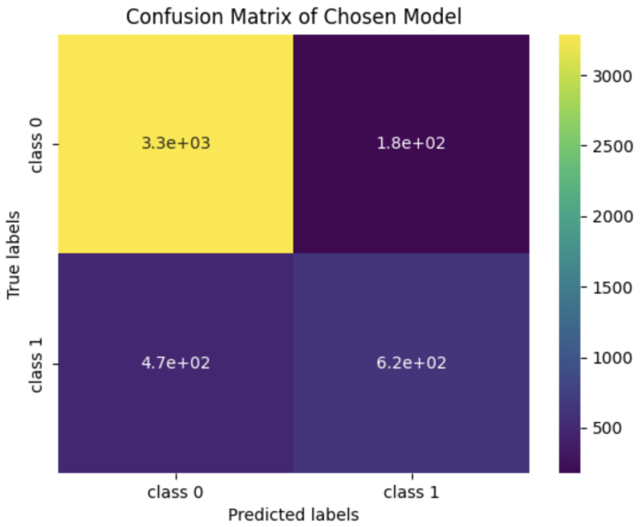
I used K-fold Cross Validation method to evaluate different models. Each time I randomly split the training data into k parts and use k-1 of them to fit a model and use the rest part to get a score of validation which is exactly the accuracy of the model. Eventually I choose the model with the highest accuracy to be my final solution model.

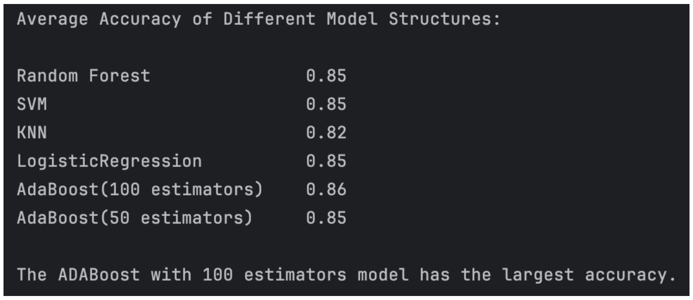
B. Confusion Matrix

I used confusion matrix with temperature to visualize the performance of the final chosen classifier.

V. Experiment

Based on the same training data and K-fold CV, I evaluated six models, Random Forest, SVM, KNN, Logistic Regression, AdaBoost with 100 estimators, AdaBoost with 50 estimators. By calculating the accuracy of these models I get the best model and train it on the training data with 0.8 portion train data and 0.2 validation data, and generate the confusion matrix. Then we can use this model to predict testlabel. The result is shown in figures.





```
Average Accuracy of Different Model Structures:

Random Forest               0.85
SVM                         0.85
KNN                         0.82
LogisticRegression          0.85
AdaBoost(100 estimators)    0.86
AdaBoost(50 estimators)     0.85

The ADABoost with 100 estimators model has the largest accuracy.
```

Fig. 1. Result of my code.

To analyze the result of experiment, we focus on the performance of different models. As shown in result, except for KNN, all other models have very similar results. It is hard to say if there is some outperformance between those models.

KNN model is sensitive to irrelevant features or not so relevant features. The Euclidean distance between identities in high dimensional space is highly hooked with the distribution of different axis. The sparser an attribute is, the bigger influence it has on the calculation of distances between identities. Yet my KNN model does not have valid method to adjust weights of different attribute. This is might the reason why KNN has a relatively poor performance. To improve this model we can loosen the distribution of more important attributes and tighten those not so important ones.

VI. Limitations and Future Solutions

There are still many possible model hyperparameter settings which may lead to even better models with higher accuracy, for example, the proportion of training data and validation data, the estimator number of Random Forest, k of KNN model, kernel function of SVM, etc. A best model should be out of thousands of adjustment of parameters and trials. To get better model, more experiments are needed.