A Comparative Study on the Adults Dataset using Different Machine Learning Methods

Yunhang Lin	Zhuoyu Feng	Xiaoyue Chen	Xintong Liu
UNI: yl4860	UNI: zf2272	UNI: xc2551	UNI: xl3121

December 15, 2022

Abstract

Income is very important for both individuals and merchants. People are eager to find a way to get rich so as to improve their lives. Enterprises want to predict the purchasing power of customers just in terms of their basic information. Therefore, in this study, we would predict adults' income by 14 types of features. It is based on the Adult dataset from UCI Machine Learning Repository. In the original paper, the author proposed NBTree to do income classification, which is a combination of decision trees and Naive Bayes. Some other strategies are also put forward in other works. In this paper, we reproduced the methodologies and results of previous works by R. On the basics of Exploratory Data Analysis, evaluate the pros and cons, as well as come up with new strategies for XGBoost and LightGBM to promote the performance of the model. The highest accuracy, recall, precision, and F1-score for the test dataset reached 0.873, 0.938, 0.993, and 0.918.

Keywords: Adult dataset, NBTree, Exploratory Data Analysis, XGBoost, LightGBM

1 Introduction

Income is always the biggest concern for people to struggle with. What factors most affect the amount of income has long been a puzzle. On one hand, education, age, and work class should have a positive relationship with income in the subconscious. So that individuals try their best to derive a higher degree and longer experience to obtain better pay. On the other hand, some other factors like race, sex, nationality, and marital status may also have potential influences on the salary. To whom concerns, these features would be fantastic elements to conclude an optimal plan for their future career development.

More critically, the consumer market has a strong need to capture the income of clients. Whether you're selling a product or a service, every commodity has a target customer group that satisfies a certain buyer persona. Income is an important consideration since it determines people's purchasing power. When doing publicity and user promotion, precise delivery to the target group of products will increase efficiency and reduce costs. For instance, high-end products need to track high-income groups for a long time due to the high profit of a single product, while low-end products may need to take the route of small profits and quick sales, so they need to sink the market on a large scale to attract low-income groups.

However, income is a very private thing for customers. Seldom people would tell a store or an APP that how much he gets every month. Therefore, how to recognize buyers' income through other information is of great importance. Merchants can obtain the necessary information of users through such as membership systems. In such a scenario, making use of these features to train a machine-learning model and do some data analysis would be a good choice for the decision of future marketing strategy. To some degree, it can be transferred into a specific classification/prediction problem, as well as the exploratory data analysis of representations.

Based on the above two scenario assumptions, We've identified potential situations where this problem can be applied. To be convenient, in our study, we first did exploratory data analysis for the dataset. Data cleaning and data integration were applied to construct a more balanced dataset. Some weird and missing values were filled. Then, several previous papers were selected and several traditional methodologies were reproduced by R, including Logistic Regression, Naive Bayes, C4.5 decision tree, C5.0 decision tree, NBTree, Bagging, and Random Forest. Finally, XGBoost and LightGBM were migrated by R to do some novel exploration. The pros and cons were analyzed.

This paper is mainly constructed in the following order. The first section is the introduction. It depicts the application area of the research subject and describes the general problems. The second section shows the description of the dataset and its attributes, as well as illustrates the review of previous works. The third section introduces the strategy and result of exploration data analysis and data preprocessing. The fourth section demonstrates the details of reproduction. The fifth section proposed our innovation methodologies. The sixth and seventh parts are the discussion and conclusion.

2 Literature Review

2.1 Dataset Description

This adult dataset was extracted from the census bureau database, coming from UCI Machine Learning Repository[6]. The donors are Ronny Kohavi and Barry Becker. By means

Table 1: Dataset Description

Data Set Characteristics:	Multivariate	**Number of Instances:**	48842
Attribute Characteristics:	Categorical, Integer	**Number of Attributes:**	14
Associated Tasks:	Classification	**Missing Values?**	Yes

of MLC++ GenCVFiles, they randomly split the dataset into train and test subsets, including 2/3 for the train and 1/3 for the test. Totally, there are 48842 instances, mixing with continuous and discrete values (train=32561, test=16281). Besides, some rows may contain missing values. Therefore, 45222 rows would be left if removing unknown values (train=30162, test=15060). 6 duplicate or conflicting instances.

It discretized gross income into two ranges with a threshold of 50,000 and converted unknown to "?". The prediction task is determining whether a person makes over 50K a year. Therefore, it is a binary classification. The target includes two parts: ($>$ 50K, $<=$ 50K). There are 14 candidate features that were prepared to complete such a prediction, including categorical and integer(numerical) attributes. The description and options show as the following.

- **age** shows the age of people. It is a set of continuous numbers, from 17 to 90 years old.
- **workclass** is a categorical attribute, which can be divided as Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
- **fnlwgt** represents the final weight. The weights on the Current Population Survey (CPS) files are controlled to independent estimates of the civilian noninstitutional population of the US. These are prepared monthly for us by Population Division here at the Census Bureau. We use 3 sets of controls. These are:

  1. A single cell estimate of the population 16+ for each state.
  2. Controls for Hispanic Origin by age and sex.
  3. Controls by Race, age, and sex.

  We use all three sets of controls in our weighting program and "rake" through them 6 times so that by the end we come back to all the controls we used. The term estimate refers to population totals derived from CPS by creating "weighted tallies" of any specified socioeconomic characteristics of the population. People with similar demographic characteristics should have similar weights. There is one important caveat to remember about this statement. That is that since the CPS sample is actually a collection of 51 state samples, each with its own probability of selection, the statement only applies within the state.

- **education** is a continuous numerical attribute showing the number of education years. It ranges from 1 to 16 years.
- **education-num** is a categorical extension of education attribute, which can be divided as Bachelor, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.
- **marital-status** is a categorical attribute, which can be divided into Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.
- **occupation** is a categorical attribute, which can be divided into Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.
- **relationship** is a categorical attribute, which can be divided into Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.
- **race** is a categorical attribute, which can be divided into White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.
- **sex** is a binary attribute, which can be divided into Female, Male.
- **capital-gain** is a numerical attribute, which is continuous.
- **capital-loss** is a numerical attribute, which is continuous.
- **hours-per-week** is a numerical attribute, which is continuous.
- **native-country** is a categorical attribute, which can be divided into United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.

2.2 Original paper review

This dataset was first cited by Ron Kohavi[10]. In this paper, it proposed a new algorithm named NBTree, which is a combination of decision-tree classifiers and Naive-Bayes classifiers. In other words, the decision-tree nodes containing univariate splits as regular decision-trees also

have Naive-Bayes classifier leaves. It not only remains the interpretability of both Naive-Byes and decision trees but also results in better performances compared with both constituents, especially in the larger dataset.

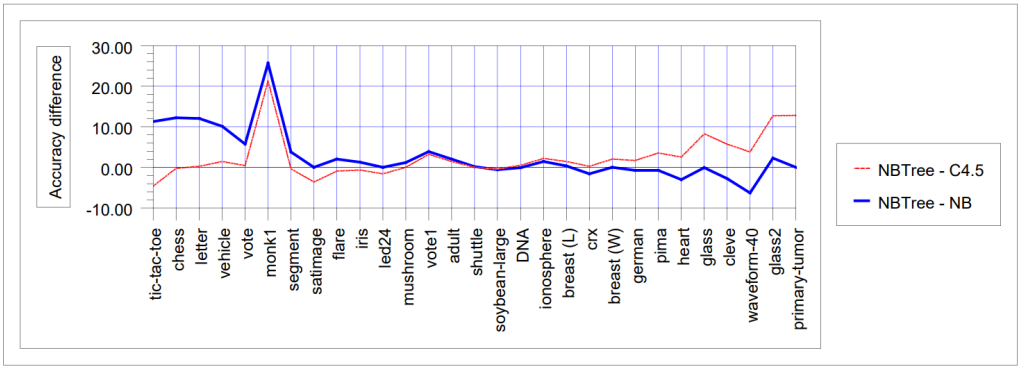


Figure 1: The accuracy differences. One line represents the accuracy difference between NBTree and C4.5 and the other between NBTree and Naive-Bayes. Points above zero show improvements. The files are sorted by the difference between the two lines so that they cross once.

The algorithm is similar to the classical recursive partitioning scheme, except that the leaf nodes created are Naive Bayes classifiers instead of predicting the nodes of a single class. A standard entropy minimization technique is used to select thresholds for continuous attributes, just like a decision tree. By discretizing the data, the utility of the node is calculated using the 5 times cross-validation accuracy estimation of Naive Bayes at the node. The utility of the split is the weighted sum of the utility of the node, where the weight given to the node is proportional to the number of instances that reach down to the node. For continuous attributes, a threshold is also found at this stage.

After getting the highest utility of certain attributes, it would check whether it is better than the utility of the current node and return. Otherwise, a Naive-Bayes classifier would be created for the node. Then, it will partition the set to the test by a threshold or a multi-way split according to whether it is continuous or not. For each child, call the algorithm recursively on the portion that matches the test leading to the child. A set of labeled instances will output a decision tree with Naive-Bayes categorizers at the leaves.

Using the Adults dataset for testing, the evaluation result is mainly a comparison among the C4.5 decision tree[11], Naive-Bayes[5], and NBTree. After the removal of unknowns from train/test sets, the error accuracy is reported as follows. The other further methodology comparisons are shown in Table 2.

- C4.5: $84.46\% \pm 0.30\%$

Table 2: Error rates of other algorithms after the removal of unknowns and using the original train/test split. All these numbers are straight runs using MLC++ with default values.

	Algorithm	Error(%)		Algorithm	Error(%)
1	C4.5	15.54	9	CN2	16.00
2	C4.5-auto	14.46	10	HOODG	14.82
3	C4.5 rules	14.94	11	FSS Naive Bayes	14.05
4	Voted ID3 (0.6)	15.64	12	IDTM (Decision table)	14.46
5	Voted ID3 (0.8)	16.47	13	Naive-Bayes	16.12
6	T2	16.84	14	Nearest-neighbor (1)	21.42
7	1R	19.54	15	Nearest-neighbor (3)	20.35
8	NBTree	14.10	16	OC1	15.04

- Naive-Bayes: $83.88\% \pm 0.30\%$
- NBTree: $85.90\% \pm 0.28\%$

Given m instances, n attributes, and l tag values, the complexity of the attribute selection phase of the discretized attribute is $O(mnl^2)$. If the number of attributes is less than $O(logm)$(which is usually the case) and the number of tags is small, then using cross-validation takes less time to select attributes than sorting instances by each attribute. Therefore, we can expect NBTree to scale well to large databases. In the paper, the author extends the evaluation to various datasets, which confirms this conclusion as shown in Fig.1.

2.3 Other related works

With this data set, other researchers are also conducting different studies in different aspects.

Zadrozny[13] formalized the problem of sample selection bias in machine learning terms and investigated analytically and experimentally how many well-known classifier learning methods are affected by it. Bianca also proposed a bias correction method, which is especially suitable for classifier evaluation under sample selection bias. Sample selection bias is solved in the context of learning and evaluating classifiers.

Rosset[12] improved the understanding and practicability of AUC as a model selection and discrimination criterion for classification scoring models. A theoretical method for calculating AUC differential moments was introduced. The conclusion was that in cases of high uncertainty, AUC may be a better performance indicator than empirical error to distinguish models, even if the ultimate goal is to classify well.

Caruana et al[1][2] compared the AUC performance of seven different supervised learning algorithms, including SVMs, neural nets, decision trees, k-nearest neighbor, bagged trees,

boosted trees, and boosted stumps. For each algorithm, they tested many different variants of the algorithm. A method for constructing sets from thousands of model libraries was proposed. The model library was generated by different learning algorithms and parameter Settings. Step forward and select the model to add performance maximization to the collection. The comprehensive selection allows for the comprehensive optimization of performance metrics such as accuracy, cross-entropy, average accuracy, or ROC region.

Kao et al[7] showed that the decomposition method with alpha seed is very effective for solving linear support vector machine sequences with more data than attributes. The motivation for this strategy is Keerthi and Lin[9], who demonstrated that for data from non-linearly separable support vector machines, the dual solution has the same free and bounded components for sufficiently large C.

Cerquides et al[3] showed that the maximum value of a posteriori TAN (Tree Augmented Naive Bayes) model can be efficiently calculated under appropriate conditions. In addition, they proved that in the posterior TAN model, the weighted set with k maximum can also be efficiently computed. This allows efficient TAN ensemble learning and calculation of model uncertainty and is used to construct two classifiers. Introduce prior knowledge of structure or parameters into the learning process.

3 Exploratory Data Analysis

3.1 Data Exploration

In the original dataset, we have 48842 instances and 15 columns, including 1 as our target variable. Among those 14 attributes, 6 are numerical, all of them are integer data, and the rest are categorical.

There are several issues with this dataset, for example, some columns have some missing values, and the target variable column shows it is a slightly imbalanced dataset. Besides, we need to some essential pre-processing steps like transforming categorical data into numerical ones, checking outliers, before we feed them into our machine learning models.

In this chapter, we will discuss those data cleaning steps as well as displaying some of the plots we made to visualize and better understand our data.

3.2 Data Cleaning

3.2.1 Factors Combining

From the structure output, we can see that some of these columns have a large number of factors. We can clean these columns by combining similar factors, thus reducing the total

number of factors.

- For the **'workclass'** column, we combined some of the values including:
  1. Combined "Without-pay" and "Never-worked" into "Unemployed";
  2. Combined "State-gov" and "Local-gov" into "SL-gov";
  3. Combined "Self-emp-inc" and "Self-emp-not-inc" into "Self-employed";

- For the **'marital-status'** column, we combined some of the values including:
  1. Combined "Married-AF-spouse", "Married-civ-spouse" and "Married-spouse-absent" into "Married";
  2. Combined "Divorced", "Separated" and "Widowed" into "Not-Married";

- For the **'native-country'** column, we combined some of the values including:
  1. Combined "Canada", "Cuba", "Dominican-Republic", "El-Salvador", "Guatemala", "Haiti", "Honduras", "Jamaica", "Mexico", "Nicaragua", "Outlying-US(Guam-USVI-etc)", "Puerto-Rico", "Trinadad&Tobago", "United-States" into "North America";
  2. Combined "Cambodia", "China", "Hong", "India", "Iran", "Japan", "Laos", "Philippines", "Taiwan", "Thailand", "Vietnam" into "Asia";
  3. Combined "Columbia", "Ecuador", "Peru" into "South America ";
  4. Combined "England", "France", "Germany", "Greece", "Holand-Netherlands", "Hungary", "Ireland", "Italy", "Poland", "Portugal", "Scotland", "Yugoslavia" into "Europe";
  5. Combined "South" and "?" into "other".

3.2.2 Dealing with Missing Data

During the data cleaning we can see that there were some values with just a "?". We can convert these values to NA so we can deal with it in a more efficient manner.

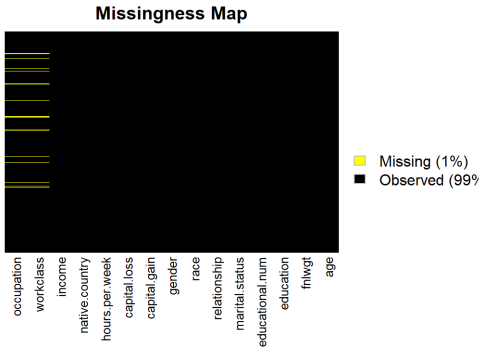
Then we created a missingness map using the 'Ameliato' library to get a visual idea of where there are NA values in the dataframe.

From the missmap, we can see that all of the NA values are found within the occupation and workclass columns. We will choose to omit these values by simply dropping those rows with missing values since there are only a few of them.

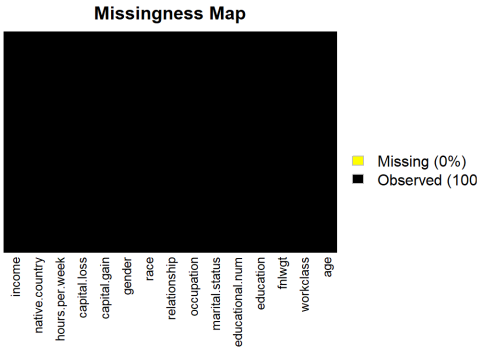
We can see that all of the NA values have been omitted from the dataset.

3.3 Data Visualization

To better understand our dataset, analyzing it in a more intuitive and interactive way, we did data visualization by using visual elements like charts and graphs, so that we could grasp difficult concepts or identify new patterns.



(a) Missingness Map 1



(b) Missingness Map 2

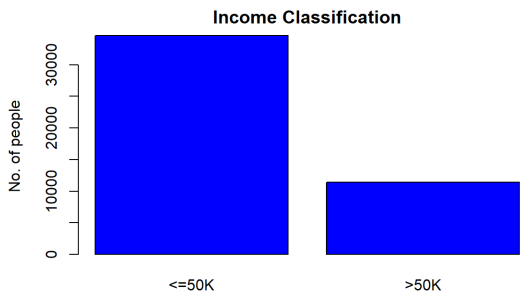
The original dataset contains a distribution of 24.08% entries labeled with >50k and 75.91% entries labeled with $\leq$ 50k. We plotted the income classification information in Figure 2(a).

Then we analyzed the relationship between income and age in this dataset. First we plotted a histogram of ages that is colored by income, as shown in Figure 2(b). As we notice majority of the people make less than <50k a year. However, we observe people earning >50k are in their mid career. We make this hypothesis based on the age. Here the coloring is indicative of percentage. From this plot we can see that the percentage of people who make above 50K peaks out at roughly 35% between ages 30 and 50. Then in Figure 2(c), we used another type of visualization - a boxplot to further capture the distribution of the data, based on a five number summary ("minimum", first quartile [Q1], median, third quartile [Q3] and "maximum").

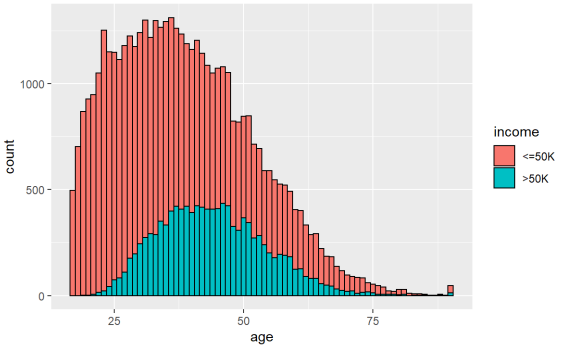
In Figure 2(d), We drew a histogram for workclass, which consists of rectangles whose area is proportional to the frequency of each workclass and whose width is equal to the class interval. This can be an approximate representation of the distribution of this feature, we can tell most samples for workclass are from the self-employed class.

Next we check the same for the column 'hours worked per week' by combining histogram and boxchart. From Figure 2(e), we can see that most people work 40 hours per week, which makes sense as it is the full-time job standard in many countries. From Figure 2(f) we may get the conclusion that more working hours per week may contribute to higher income. The same thing for educational num, as shown in Figure 2(g).

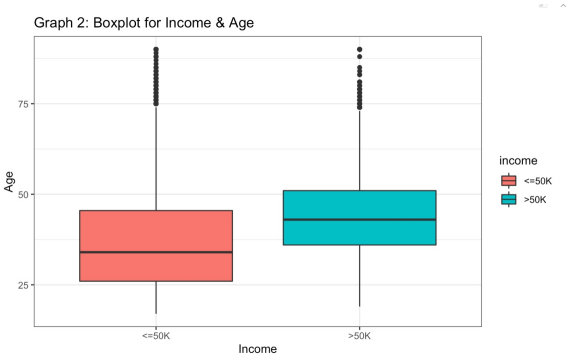
At last, we analyzed the relationship between the income and native continent in Figure 2(h). It seems almost all the data samples are from North America, within which over 2/3 people receive income less than 50k. This also indicates that this dataset may be biased as it focuses more on the people from North America.



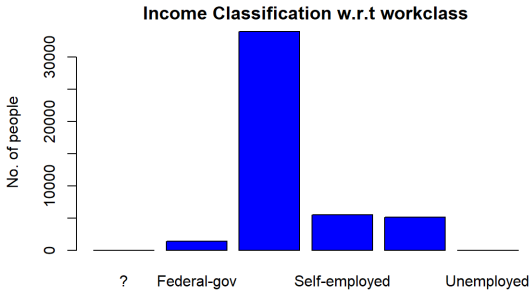
(a) Income Classification



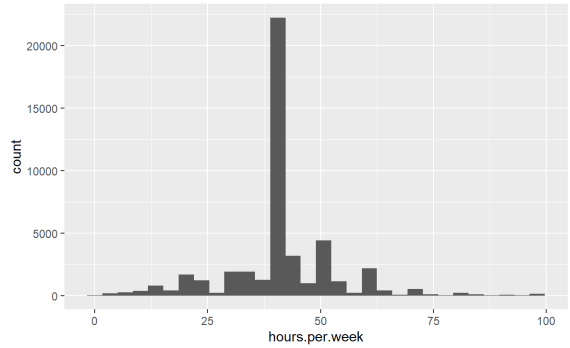
(b) Income vs Age



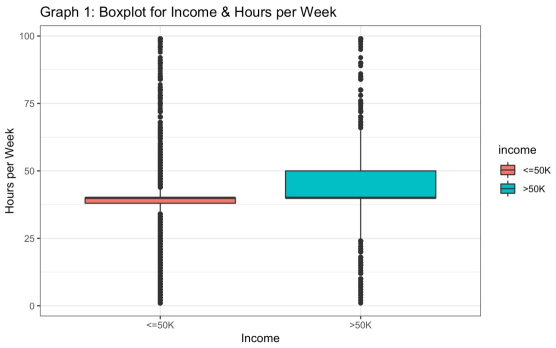
(c) Boxplot for Income vs Age



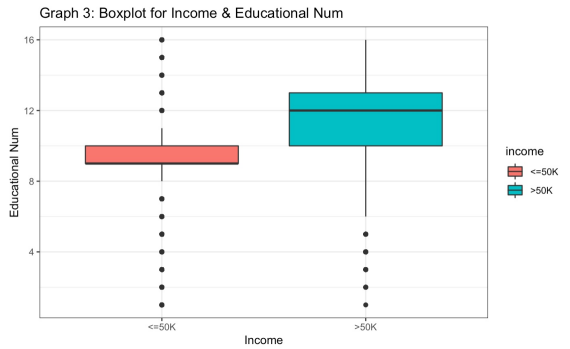
(d) Histogram for Workclass



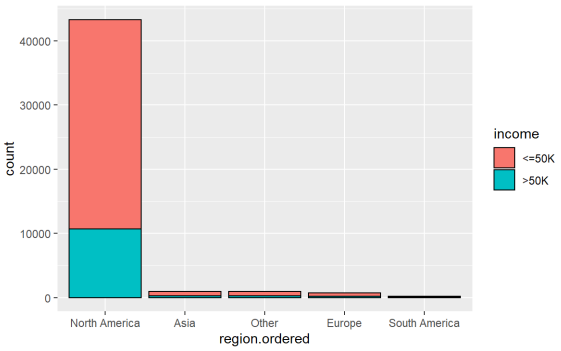
(e) Hours Worked per Week



(f) Boxplot for Income vs Hours Worked per Week



(g) Boxplot for Income vs Educational Num



(h) Income vs Native Continent

Figure 2: Data Visualization

4 Reproduction

The paper that we are trying to reproduce introduces that the algorithm of naive-bayes can achieve a surprising accuracy in the task of classification [10]. However, the previous indications are that the naive-bayes algorithm can work well only in small datasets. In large-scale datasets, the naive-bayes algorithm does not work as well as the decision tree. Under this observation, previous research introduces a new algorithm called NBTree. The functionality of this algorithm involves both the decision tree classifier and the naive-bayes classifier. Specifically, the node of the decision tree of the NBTree classifier is split in the same method as the regular decision tree, but the difference is that the leave of it contains the naive-bayes classifier. Thus, it seems like a combination of the decision tree classifier and the naive-bayes classifier. Since the NBTree classifier include the interpretability of both classifiers, the previous experiment has concluded that it can outperform both of the algorithms. Also, the NBTree algorithm also works well under the condition of large-scale datasets.

In our experiments, we are trying to reproduce this paper by using the programming language R. The paper also uses a decision tree model C4.5 to compare the training results with the naive-bayes. The model training of our experimental simulation that consists of naive-bayes, C4.5, and NBTree are implemented under the adult datasets. The adult datasets are split into training datasets and testing datasets by ratios of 0.7 and 0.3. The model is trained firstly by the training datasets, then using the testing datasets to predict the model and collect result data. The experimental results collect the accuracy value, recall value, precision value, and the confusion matrix of all the models, which can be referred to in table 3. Based on the result of testing accuracy, in which naive-bayes can achieve 80.7% testing accuracy and 83.3% for the NBTree model, we can conclude that the NBTree classifier can reach a better learning result than the naive-bayes classifier.

Not only the models naive-bayes, C4.5, and NBTree that are mentioned in the reproduced paper, in our experiment, we also collect the model training results of other models which consist of logistic regression, C5.0, bagging, and random forest. All these models are trained under the datasets of adults. In order to demonstrate the performance of each model more explicitly, we also train all of these 7 models under different amounts of datasets. The number of datasets we used to train these models started from 2500 to 45000. The plot of the final result of accuracy can be referred to Fig.6, in which we can conclude that model C5.0 and random forest can perform the best training outcome over these models. And all these models can outperform the naive-bayes classifier.

To analyze the datasets, we also use the model results to demonstrate the distribution of different attributes of adult datasets. Referred to Fig.3, which illustrates the distribution of attributes with an integer value, such as age, fnlwgt, educational number, capital gain, capital

loss, and hour per week. For example, for the distribution of age attribute, the plot indicates that most of the adults with income less than 50K are around the age of 25, and most of the adults with income more than 50K are with age around 40.

We will discuss the logic of those 7 models and the experimental results more explicitly in the next section.

4.1 Model Introduction

4.1.1 Logistic Regression

Logistic Regression is a "Supervised machine learning" algorithm that can be used to model the probability of a certain class or event. It is used when the data is linearly separable and the outcome is binary or dichotomous in nature. That means Logistic regression is usually used for Binary classification problems.

We can call a Logistic Regression a Linear Regression model but the Logistic Regression uses a more complex cost function, this cost function can be defined as the 'Sigmoid function' or also known as the 'logistic function' instead of a linear function.

4.1.2 Naive Bayes

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

4.1.3 C4.5

C4.5 is an algorithm used to generate a decision tree developed by Ross Quinlan. A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

C4.5 decision tree is a modification over the ID3 Decision Tree. C4.5 uses the Gain Ratio as the goodness function to split the dataset, unlike ID3 which used the Information Gain.

4.1.4 C5.0

The C5.0 algorithm has become the industry standatd for producing decision trees, because it does well fo rmost types of problems directly out of the box. Compared to more advanced and sophisticated machine learning mnodels (e.g. Neural Networks and Support Vector Machines), the decision trees under the C5.0 algorithm generally perform nearly as well but are much easier to understan and deploy.

C5.0 uses the concept of entropy for measuring purity. The entropy of a sample of data indicates how mixed the class values are; the minimum value of 0 indicates that the sample is completely homogenous, while 1 indicates the maximum amount of disorder.

4.1.5 NBTree

The NBTree is a hybrid classifier of decision tree and naïve Bayes classifiers. In NBTree nodes contain and split as regular decision tree, but the leaves are replaced by naïve Bayes classifier.

4.1.6 Bagging

Bagging, also known as bootstrap aggregation, is the ensemble learning method that is commonly used to reduce variance within a noisy dataset. In bagging, a random sample of data in a training set is selected with replacement—meaning that the individual data points can be chosen more than once.

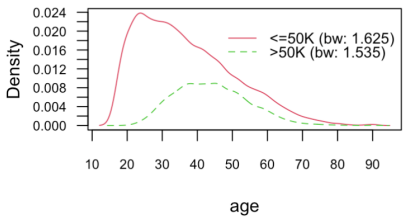
After several data samples are generated, these weak models are then trained independently, and depending on the type of task—regression or classification, for example—the average or majority of those predictions yield a more accurate estimate.

4.1.7 Random Forest

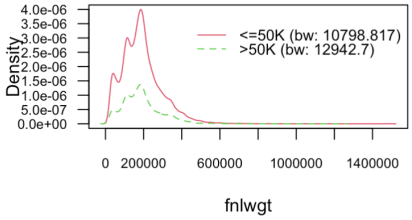
Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. It uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree.

4.2 Experimental Results

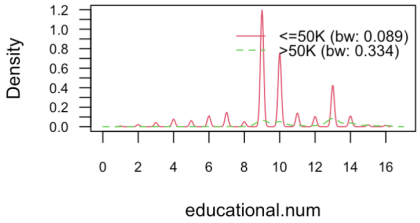
For the experimental results, we train 7 models under the adult datasets and collect the training result that consists of confusion matrix, accuracy, recall, and precision of both training and testing datasets. The models we train under adult datasets consist of Naive-bayes, C4.5, C5,0, NBTree, logistic regression, bagging, and random forest. We also compare the plot of the accuracy of the original paper and our reproduced work to analyze the feasibility of our results. Moreover, in order to compare the testing accuracy of all of the models, we create a plot of accuracy vs instance over all models, in which the model is trained under different amounts of datasets.



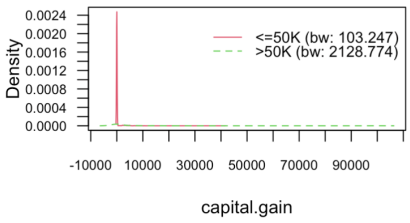
(a) age



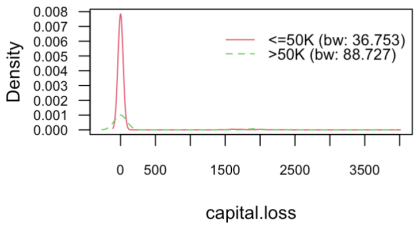
(b) fnlwgt



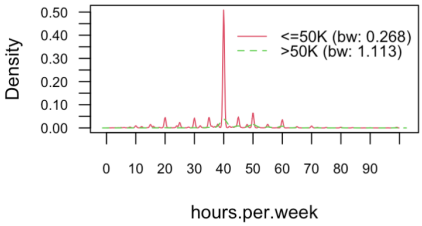
(c) educational.num



(d) capital.gain



(e) capital.loss



(f) hours.per.week

Figure 3: Visualization of attributes in datasets

	<=50K	>50K
<=50K	10295	88
>50K	2565	862

(a) Naive Bayes

	<=50K	>50K
<=50K	9732	651
>50K	1472	1955

(b) C4.5

	<=50K	>50K
<=50K	8858	1525
>50K	770	2657

(c) NBTree

	FALSE	TRUE
<=50K	9589	794
>50K	1358	2069

(d) Logistic Regression

	<=50K	>50K
<=50K	9803	580
>50K	1340	2087

(e) C5.0

	<=50K	>50K
<=50K	9558	825
>50K	1287	2140

(f) Bagging

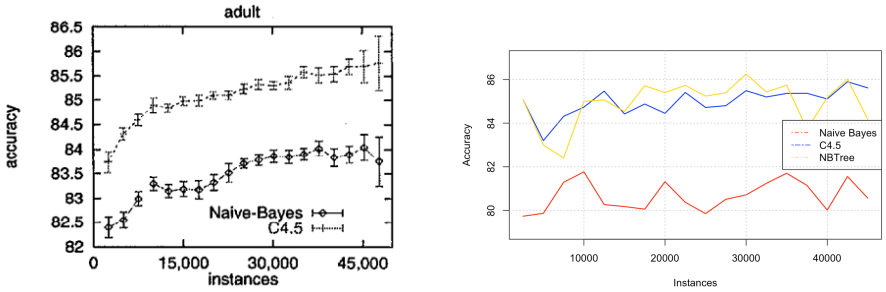
	<=50K	>50K
<=50K	9727	656
>50K	1325	2102

(g) Random Forest

Figure 4: Confusion matrix of all models

Table 3: Final results of models

Training Data				
Model	Accuracy	Recall	Precision	F1-score
Logistic Regression	0.849	0.879	0.927	0.902
Naive Bayes	0.809	0.800	0.995	0.887
C4.5	0.879	0.889	0.959	0.923
C5.0	0.877	0.891	0.954	0.921
NBTree	0.840	0.923	0.858	0.889
Bagging	0.996	0.996	0.999	0.997
Random Forest	0.947	0.951	0.981	0.966
Testing Data				
Model	Accuracy	Recall	Precision	F1-score
Logistic Regression	0.843	0.873	0.926	0.899
Naive Bayes	0.807	0.799	0.993	0.885
C4.5	0.846	0.868	0.937	0.901
C5.0	0.861	0.879	0.944	0.910
NBTree	0.833	0.920	0.853	0.885
Bagging	0.847	0.881	0.921	0.901
Random Forest	0.857	0.880	0.937	0.908



(a) Paper Results

(b) Reproduction

Figure 5: Accuracy vs Instances: Comparison of paper and reproduced results

According to Fig.4, it is the confusion matrix of every model after training. The confusion matrix is used to compute the result values of accuracy, recall, and precision. For instance, for the model of Naive-bayes, 10295 samples that income less than 50K are classified into the group of less than 50K correctly. And 862 means that this amount of sample that income greater than 50K is classified into the group of greater than 50K correctly. And the other two value is the samples that are classified into the incorrect group. Thus, we can use this confusion matrix to compute the accuracy of prediction, which is the sum of the sample that are classified correctly over the total number of testing datasets. The confusion matrix can also be used to compute the result values of recall and precision.

After obtaining the confusion matrix of all models, we implement the mathematical computation over these confusion matrices to compute the accuracy, recall, and precision of all models. According to table 3, it demonstrates the result values of all models. Naive-bayes, C4.5, and NBTree are the models that are discussed in the original paper. We can observe that based on the prediction results under testing datasets, Naive-bayes can achieve 80.7% accuracy, C4.5 is 84.6 % accuracy, and NBTree obtains 83.3% accuracy. The accuracy of C4.5 and NBTree are performed better than the model of naive-bayes. These results indeed demonstrate that the naive-bayes algorithm does not perform as well as the decision tree (C4.5) under large-scale datasets, which is the statement indicated in the original work. And the introduced algorithm NBTree also can work better than the model of naive-bayes under large-scale datasets.

Besides the training results of models that mentioned in the original paper, we also involve some other models to compare their model learning outcomes. Based on the results data, we observe that C5.0 and random forest can achieve the best training outcome, in which the testing accuracy of random forest is 85.7%, and C5.0 can obtain 86.1% accuracy. Also, other models can also perform better than the model of naive-bayes under the condition of large-scale datasets.

In order to conclude the feasibility of our experimental results, we plot a graph of accuracy

vs instance (Fig.5) to demonstrate the comparison of results from the original paper and our reproduced work. Part (a) of this figure is the plot from the original paper, and part (b) is the experimental results. In reproduction, The blue line is the result of C4.5, and the red line is the result of naive-bayes. It is easy to observe that the model of C4.5 can outperform around 3% to 4% accuracy better than the model of naive-bayes along with different numbers of training datasets. The paper results also demonstrate that the model of C4.5 performs better than the model of naive-bayes. And there is also around 3% to 4% accuracy in difference. Moreover, in the graph of reproduction, the yellow line represents the results of the NBTree model, which is the new model introduced in the original paper. It also illustrates that the NBtree model can perform better than naive-bayes. And in some cases of dataset amount, NBTree can perform better than the decision tree C4.5. This is because the functionality of NBTree is the hybrid of a decision tree and naive-bayes. And under different scaling of dataset amount, the performance of these models and the comparison of their results would be different. Therefore, our experiment successfully reproduces the previous work under the same models and the same large-scale datasets.

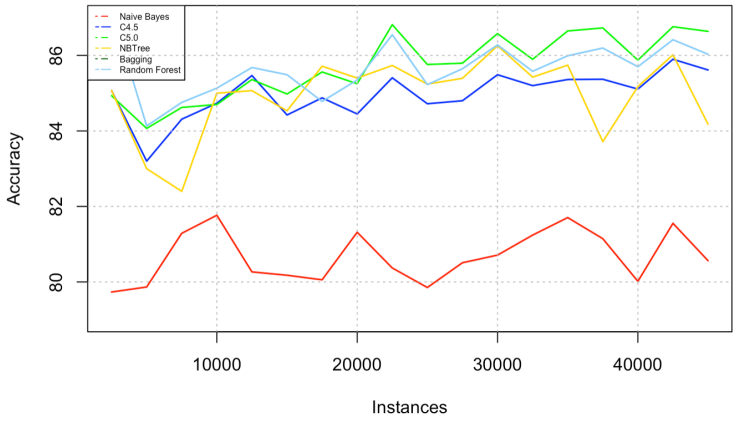


Figure 6: Accuracy vs Instances: All models

Finally, to better analyze the training results of all 7 models, we also plot the figure of accuracy vs instances of all models (Fig.6). Based on this plot, it is easily demonstrated that all of the models can perform better than naive-bayes over different amounts of datasets. Especially for the model of C5.0 and random forest, these 2 models can perform approximately better than other models in the condition of large-scale datasets.

In conclusion, our experiment successfully reproduces the precious work base on the original paper of the NBTree model. In the next section, we will introduce two more models that are trained by using adult datasets.

5 Innovation

We applied the XGBoost and Light GBM models on the adult dataset as well.

5.1 XGBoost

XGBoost is a scalable end-to-end tree-boosting system. XGboost can effectively improve the computation speed and the model performance according to research studies. The two novel algorithms behind the XGBoost model are the Weighted Quantile Sketch algorithm and the Sparsity-aware Split Finding algorithm. These two algorithms are both used to find the splits in a tree. Weighted Quantile Sketch algorithm makes use of *provable theoretical guarantee* to handle the weighted data.[4].

We preprocess the data by using one-hot encoding which can make the data sparse. According to the paper [4], the XGBoost model works well for sparse data. In the experiments, XGboost has a good performance on the adult dataset, which is consistent with the paper [4].

5.2 LightGBM

Unlike other gradient boosting frameworks, LightGBM uses a leaf-wise tree growth algorithm. Moreover, Unlike XGBoost, LightGBM doesn't need to scan all the data to estimate the possible split points. Therefore, LightGBM usually converges faster. LightGBM leverages two algorithms to accelerate the training and testing: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB)[8]. The GOSS algorithm is a sampling method that retains the samples with a large gradient while doing random sampling on the samples with a small gradient. In order to retain the original distribution, LightGBM amplifies the contribution of the samples with a small gradient. Considering the fact that machine learning problem generally involves high dimensional and sparse data, some features are exclusive to each other. EFB algorithm bundles those features together. These two algorithms enable LightGBM a faster computational performance.

5.3 Experimental Results

We train XGBoost and LightGBM models under the adult datasets and collect the training result that consists of the confusion matrix, accuracy, recall, and precision of the testing datasets.

Figure 7 is the confusion matrix obtained in XGBoost and LightGBM models. As in Figure 7, we can see both the LightGBM and XGBoost perform better in Predicting 0. We use the confusion matrix to compute the accuracy, recall and precision, and F1 score of each model.

```
          Reference
Prediction    0    1
         0 6967  774
         1  464 1563
```

(a) XGBoost Confusion Matrix

```
          Reference
Prediction    0    1
         0 6953  843
         1  478 1494
```

(b) LightGBM Confusion Matrix

Figure 7: Confusion Matrix

Table 4: Metrics in XGBoost and LightGBM models

Testing Data				
Model	Accuracy	Recall	Precision	F1-score
XGBoost	0.873	0.938	0.900	0.918
LightGBM	0.865	0.936	0.892	0.913

The table 4 is the Accuracy, Recall, Precision, and F1-score metrics obtained in XGBoost and LightGBM models. According to table 4, the XGBoost model performs slightly better in this classification task. Both the accuracy in XGBoost and LightGBM models are above 86%. The Recall, Precision metrics in both XGBoost and LightGBM reach or exceed 90%. Considering the fact that the accuracy can become unreliable when the class is imbalanced, we also compute the F1-score to evaluate the model. The F1-score is the harmonic mean of the precision and recall. The best F1-score value is 1. The higher value of the F1-score, the better performance a model has. In our experiment, the f1-scores in both the XGBoost and LightGBM models are above 90% while the F1-score of the XGBoost model is slightly higher than that of the LightGBM model.

Besides calculating the metrics above, we also evaluate the relationship between the accuracy and the number of instances. We feed the model with a dataset size from 2500 to 45000. As in figure 8, when the size of the dataset is large, the accuracy of these two models is nearly the same. However, the XGBoost model reaches higher accuracy earlier than the LightGBM model.

5.4 Compare the Results

According to the metrics obtained above, both XGBoost and LightGBM perform well in our experiment. However, according to the table 4, the metrics in XGBoost models are higher than in LightGBM. And the XGBoost reaches high accuracy earlier than the LightGBM model. This may be because the LightGBM is a leaf-wise growth algorithm, which can lead to more loss reduction. This may result in overfitting the training data. Therefore, compared to

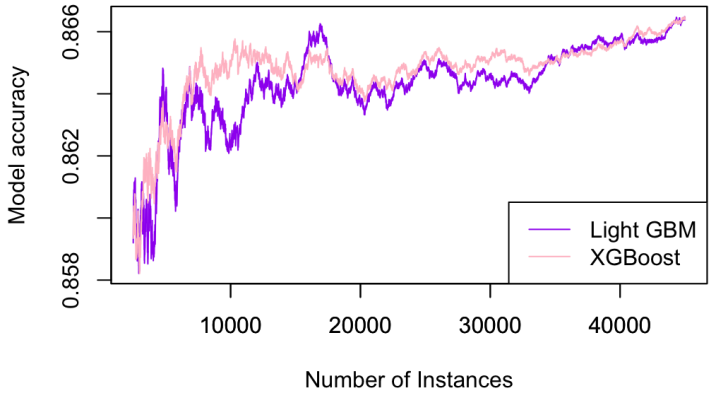


Figure 8: Accuracy vs Instances

LightGBM, XGBoost is capable to build more robust models. The results of the experiment are consistent with the research.

Compared to the models in the paper we reproduced, the accuracy and recall of XGBoost and LightGBM are higher than most of the accuracy and recall of models in the paper. The following are the detailed comparisons:

The accuracy of XGBoost is the highest among all the models we tried while the accuracy of LightGBM is the second high among them. These two models have better performance than the models in the paper.

Since the data is not well-balanced, we also need to compare the recall and precision The precision of models in the paper is higher than the precision of LightGBM and XGBoost while the recall of XGBoost and LightGBM is higher. To combine the results of precision and recall, we compare the F1-score of those models: The F1-score of the XGBoost is the highest among all the models we built and the F1-score of the LightGBM is the second high. Therefore, we can conclude that the XGBoost and LightGBM perform better than the models in the paper.

6 Conclusion

In this study, 7 models were reproduced, including Logistic Regression, Naive Bayes, C4.5 decision tree, C5.0 decision tree, NBTree, Bagging, and Random Forest. According to the corresponding papers, we worked along with their theories and strategies, then implemented the algorithms and results by means of R.

Besides, two innovation methodologies were proposed. On the basis of the Adult dataset, which contains a little bias, XGBoost and LightGBM perform well after we executed several data preprocessing. From the perspective of evaluation metrics, XGBoost shows the best accuracy, recall, and F1-score values. Respectively 87.3% accuracy, 93.8% recall and 0.918

F1-score. Those are generally much better than the models from papers. The reasons are already demonstrated in section 5.4.

7 Future work

There are still several parts of future works worth exploring.

First, bias still exists in the original dataset as shown in Section 3.3. This imbalance is reflected in the target, but also in part of the features. Potentially, it may lead to different performances between train and test sets and reduce the generalization of the model. Such bias also depicts in the difference value of recall and precision. Since the number of the positive samples ($<= 50K$) is much larger than the negative one ($> 50K$), the model gets better training for positive inputs. So that in most cases, precision is better than recall. We can augment or truncate the dataset in order to keep each classification class with equal percentages. Remove or merge minority features to keep them balanced.

Second, the neural network can be used to further improve the prediction. Based on specific input data definition and model hiding layer construction, this binary classification task may get a better result when using neural networks.

Third, the model ensemble can be used to promote the results. For example, we have already implemented so many models to classify the targets. By means of a voting strategy, we can synthesize the predictions of all the models. To some degree, it may compromise the drawbacks of each other than do a better job. Theoretically, NBTree is also a combination of Naive Bayes and decision trees from the aspect of the model.

References

[1] Rich Caruana and Alexandru Niculescu-Mizil. "An Empirical Evaluation of Supervised Learning for ROC Area." In: *ROCAI*. 2004, pp. 1–8.

[2] Rich Caruana et al. "Ensemble selection from libraries of models". In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 18.

[3] Jesús Cerquides and Ramon Lòpez de Màntaras. "Maximum a posteriori tree augmented naive Bayes classifiers". In: *International Conference on Discovery Science*. Springer. 2004, pp. 73–88.

[4] Tianqi Chen and Carlos Guestrin. "XGBoost: A scalable tree boosting system". In: *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2016, pp. 786–794.

[5] James Dougherty, Ron Kohavi, and Mehran Sahami. "Supervised and unsupervised discretization of continuous features". In: *Machine learning proceedings 1995*. Elsevier, 1995, pp. 194–202.

[6] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017. URL: `http://archive.ics.uci.edu/ml`.

[7] Wei-Chun Kao et al. "Decomposition methods for linear support vector machines". In: *Neural Computation* 16.8 (2004), pp. 1689–1704.

[8] Guolin Ke et al. "Lightgbm: A highly efficient gradient boosting decision tree". In: *Advances in neural information processing systems* 30 (2017).

[9] S Sathiya Keerthi and Chih-Jen Lin. "Asymptotic behaviors of support vector machines with Gaussian kernel". In: *Neural computation* 15.7 (2003), pp. 1667–1689.

[10] Ron Kohavi et al. "Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid." In: *Kdd*. Vol. 96. 1996, pp. 202–207.

[11] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[12] Saharon Rosset. "Model selection via the AUC". In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 89.

[13] Bianca Zadrozny. "Learning and evaluating classifiers under sample selection bias". In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 114.