COMP5318 Assignment 2

Adult Income Classification

Qinglong Zeng (470147746)

Xintong Chen (470138674)

1. Abstract

This project will use four machine learning algorithms to predict a person's income based on his/her features such as education level and working class. The four algorithms are support vector machine (SVM), stochastic gradient descent (SGD), multi-layer perceptron (MLP) of neural network and bagging decision tree(B-DT). The four algorithms are trained by a well pre-processed dataset and make prediction, their performances are evaluated and compared. Among the four algorithms, B-DT achieves the best prediction accuracy which is up to 96.4%, while the others are of around 90% accuracy.

2. Introduction

A person's income is always put down to their education level, marital status, working class and so on, how the combinations of these features affect person's income level is the problem to address in this project. A dataset of over 32500 persons' 14 features are considered under machine learning classification techniques to identify their relation to income level. A dataset of over 16200 persons' income level are to be predicted based on what the classification have learned. Income is not the only way to measure a good life, but it might reflect some of the qualities that are qualified to own the good one. Thus, by learning how these features or qualities affect people's achievement in terms of income, one can prepare his/her self on these qualities to pursue a better life.

3. Previous Work

The dataset used in this project is a quite old one, there are some good researches on it

already. In [1], the authors introduced a hybrid method combining Naïve-Bayesian classifier with Decision Tree (NBTree) to utilize the advantages of both algorithms. In this classification method, decision tree was used to segment data into branches and Naïve-Bayesian was applied on each branch. The NBTree archived an accuracy of 84.47%. Authors in [2], did a series of data exploration to learn the pattern and statistical features of the dataset, which made a good visualization on different attributes, and help to select algorithms accordingly. They then applied a bunch of algorithms on the dataset for comparison. The results showed that the Gradient Boosting Classifier made the best shot with accuracy of 86.4% on training set and 86.3% on the test set which was also the most reliable one. Most algorithms in the paper performed much worse on test set than train set without extracting features from original dataset. However, [3] applied great feature extraction and reduction to figure out how each feature and different combination of several features (not all features) affected the income. On top of this, Naïve Bayes, Logistic Regression and Decision Tree were applied for classification.

4. Classification Method

There are four different machine learning algorithms that are used in this project, they are Support Vector Machine (SVM), Stochastic Gradient Descent (SGD), Neural Network and Bagging Decision Tree. The input dataset format of these algorithms are nearly the same, so same data pre-processing procedures will be implemented for each of the algorithms.

4.1 Data Pre-processing

4.1.1 Data clean

- The column “income” have two categories, namely “<=50K” and “ >50K”, they are converted to numerical value 0 and 1 respectively
- The column “education_num” is an equivalence of the column “education”, thus, the former is dropped
- There are signs “?” in column “workclass” and “occupation” which might represent unclear or missed information, they are replaced by “nan”.

4.1.2 Transformation of categorical features

There are 8 columns with categorical attributes such as “workclass”, “sex” and “marital

status", it is essential to convert them into numerical attributes so that they can be fed into classification algorithms.

The standard to convert a category in an attribute to a number is by calculating how it is relevant to the income "greater than $50K". The more likely the samples with this category of feature have income greater than $50K, the higher numerical value the category is assigned. In this manner, the numerical value that a feature is assigned is the proportion of samples whose income is greater than 50K with this feature. Take "education" attribute for example, there are 16 categories like "Doctorate", "Bachelors" and "Masters" and so on, 74.1 percent of samples with "Doctorate" level of education have income "greater than 50K", so "Doctorate" is assigned to 0.741. Some of the results are visualized in Fig 1, Fig 2 and Fig 3 (need to zoom in to identify labels).

After converting all categorical attributes to numerical ones, these attributes can reflect the relevance to the income and their values reveals the exact extent of such correlation.

4.1.3 Scaling and transformation of numerical features

There are 5 columns with numerical attributes, namely, "age", "fnlwgt", "capital_gain", "capital_loss" and "hours_per_week". The values of some of these attributes are sparsely attributed, and the range of values in different attributes are not in the same order of magnitude. These can affect performance of classification algorithms.

In order to scaling numerical attributes, the values in each column are compressed into certain number of bins which combines a certain range of values into one section. The section is then assigned with a new value which follows exactly the same manner in transformation of categorical features. In other words, each section of values is treated as a category, and are assigned by the same values according to their relevance to income.

The three steps of data processing have provided a high-quality feature extraction for upcoming classification algorithms.

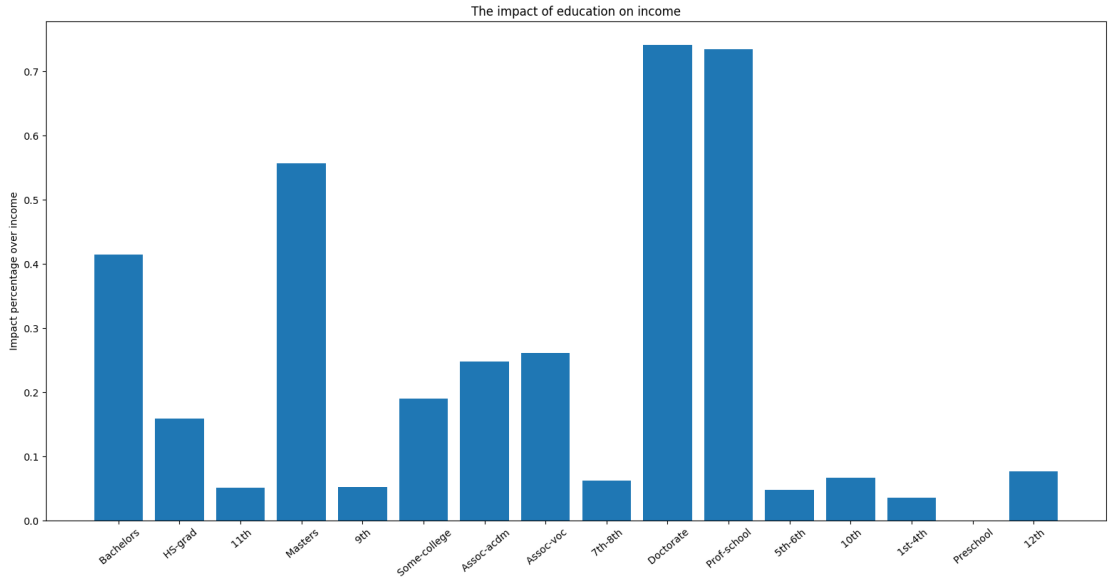


Fig 1. The impact of “education” over income level

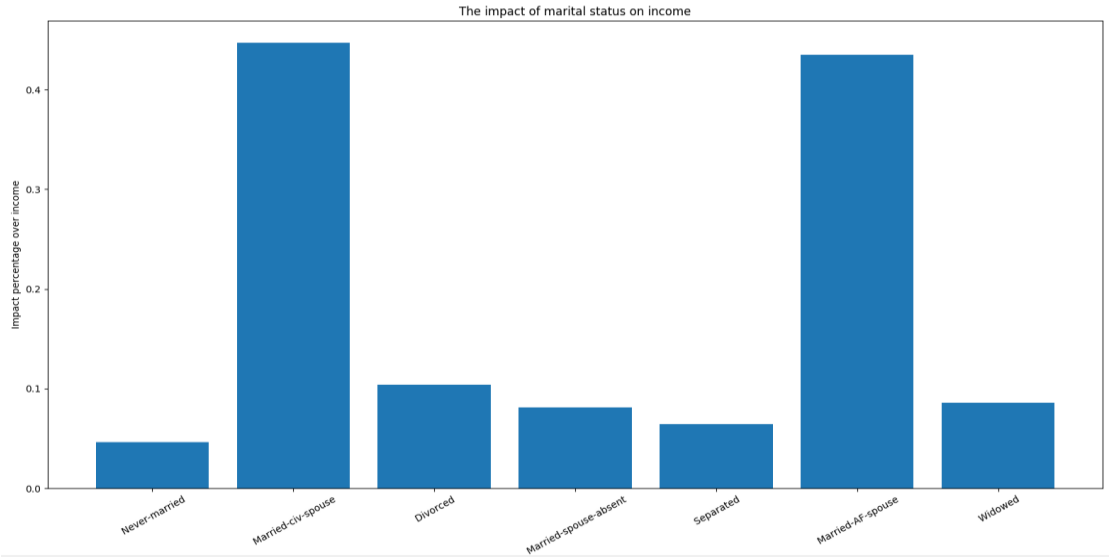


Fig 2. The impact of marital status over income level

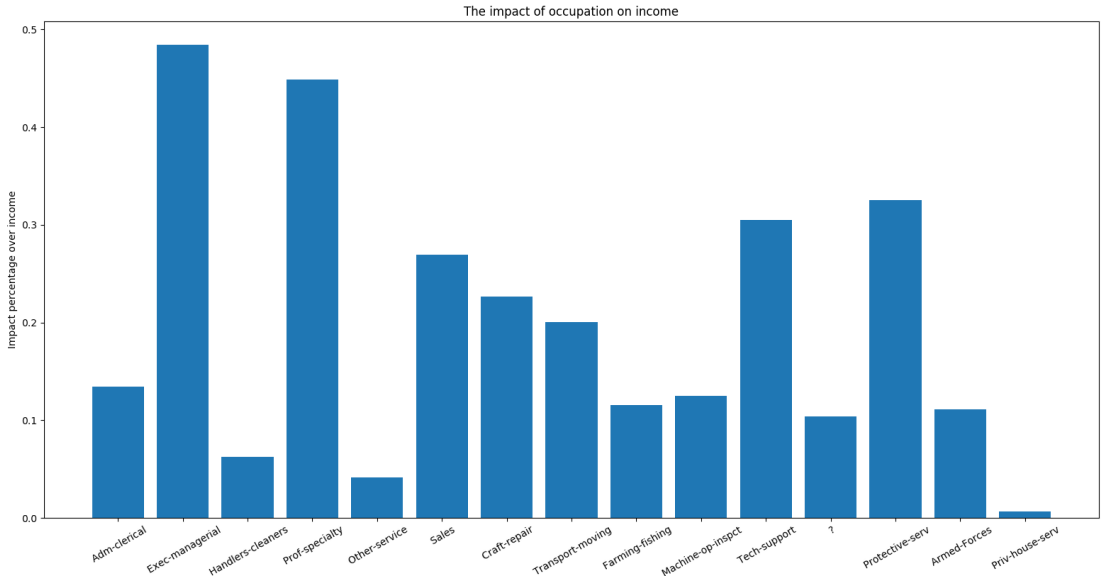


Fig 3. The impact of different occupation over income level

4.2 Algorithms Design

Grid search is applied to each of the four algorithms to pick up the most fitted parameters, and use algorithms with tuned parameters to operate on test data. The results are shown behind each algorithm.

Fine Tuning followed the scikit-learn function sklearn.model_selection.GridSearchCV. The input parameters are estimator object and a list of parameters to be used for fine tune. For each module, a series of parameters and possible values will be tested. And for the result, two overall best parameter sets correspondingly for precision and recall will be found. And in the following steps, each of these parameter sets will be substituted in test set prediction and chose the one obtains best performance as final parameter set.

Each accuracy value for every parameter set is indicated using 10-fold cross validation accuracy, and a normalized confusion matrix based on testing using test set.

4.2.1 Support Vector Machine (SVM)

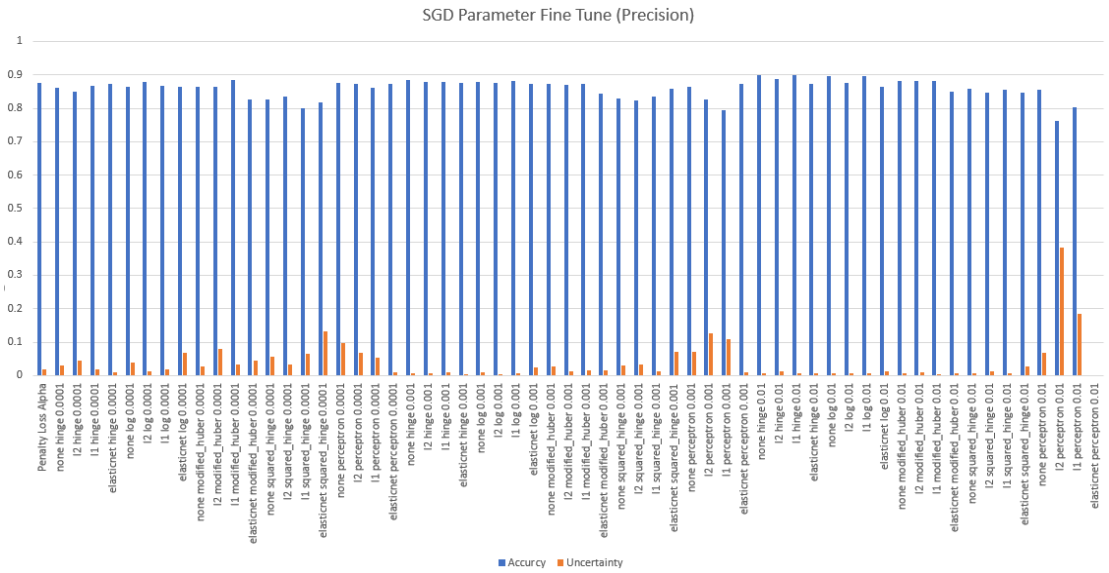
SVM is selected because it is effective in high dimensional spaces and memory efficient. There are two main parameters in C-Support Vector Classifier, the one is penalty parameter “C” of the error term which determines the margin size of hyperplane used

for classification, the larger the "C" is, the smaller the minimum margin between hyperplane and different classes is, the other is "kernel function" which is a similarity function used for easy computation. After a simple grid search on the combination of the parameters based on the prediction precision over test data, "C" is set to 10 and radial basis function (rbf) is used as "kernel function".

4.2.2 Stochastic Gradient Descent (SGD)

SGD is simple but quite efficient approach to fit linear model. It finds optimal solution using loss functions and different penalties, which are two main parameters in this classifier. The sorted feature matrix of the dataset after being preprocessed in this project follows the same standard – "the higher a feature affects income, the higher the value it is assigned to", thus they are expected to be linear after sorted but not necessary, SGD is selected as a benchmark method which takes the minimum running time and a tool to justify linearity of feature matrix.

SGD classifier holds 17 parameters and 2 attributes, in this study, loss, penalty and alpha parameter is involved in fine tune, due to these 3 parameters is possible be multiple values which is considered possibly effect classification.



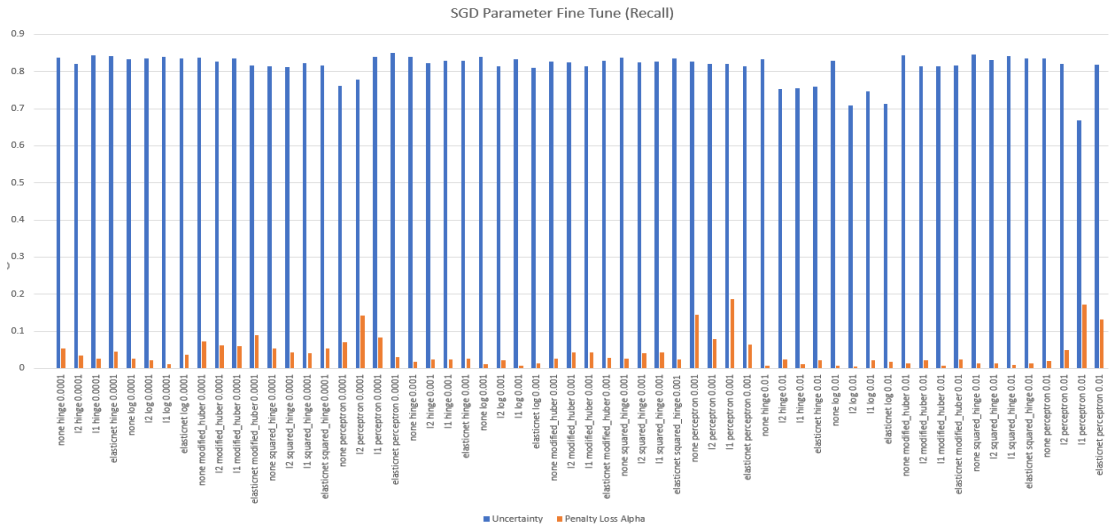


Fig 4. SGD parameter fine tune based on precision and recall

The program output with a parameter fine tune part. The fine tune part indicated to obtain corresponding precision and recall for a series parameter sets. And therefore, to obtain a best parameter set that obtains the best precision and recall. Fig 4 shows the fine tune result for SGD classifier, and vertical of these two bar charts represents precision and recall for each parameter set, respectively.

A trend of influence of each parameter on SGD performance can be seen by uncertainty. The change of uncertainty along with the change of parameter set can be seen a ramp pattern. A supposing of us for this phenomenon is that these selected parameters influences the boundary slightly and therefore effects the uncertainty. However the specific reason for this problem is not specified.

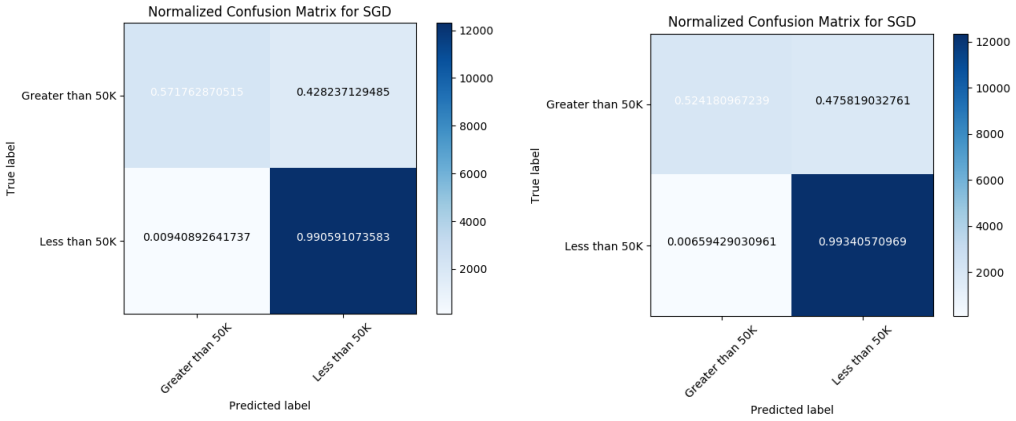


Fig 5. Normalized confusion matrix for parameter sets for SGD obtains best precision(left) and recall(right)

According to fine tune result shown in Fig 4, a parameter set {'alpha': 0.01, 'loss': 'hinge', 'penalty': 'elasticnet'} for SGD obtains the best precision. Applying this parameter set to SGD training progress. A normalized confusion matrix can be seen in Fig 5 left plot. And the corresponding result is:

Accuracy: 0.8916528468767275,

Precision: 0.9157720819443131,

Recall: 0.7811769720487252,

Fscore: 0.8234555634239179.

According to fine tune result shown in Fig 4, a parameter set that obtains the best parameter set for recall for this classifier is {'alpha': 0.0001, 'loss': 'perceptron', 'penalty': 'elasticnet'}, and the classification result is shown in Fig 2 right plot. And corresponding result:

Accuracy: 0.8825624961611694,

Precision: 0.9159437240316503,

Recall: 0.7587933384645398,

Fscore: 0.8032499992162682.

4.2.3 Multi-Layer Perceptron (MLP) of neural network

Neural Network (NN) is a kind of compute-intensive method with more parameters to tune to achieve great performance, it can handle more complex dataset with unknown pattern. Our data is not that complex, and even cleaner after feature sorting, the activation function of NN includes a method – “logistic regression” which is known as a good approach to solve binary classification. MLP of NN can combine numerous “logistic regression” together with different structure to perform learning process. Thus, it was selected to see whether a complex model can solve relative easy problem.

The same fine tune technique were applied to MLPC.

Fig 6. MLPC parameter fine tune based on precision and recall

The corresponding normalized confusion matrix for precision and recall is shown in Fig 7 on left and right respectively.

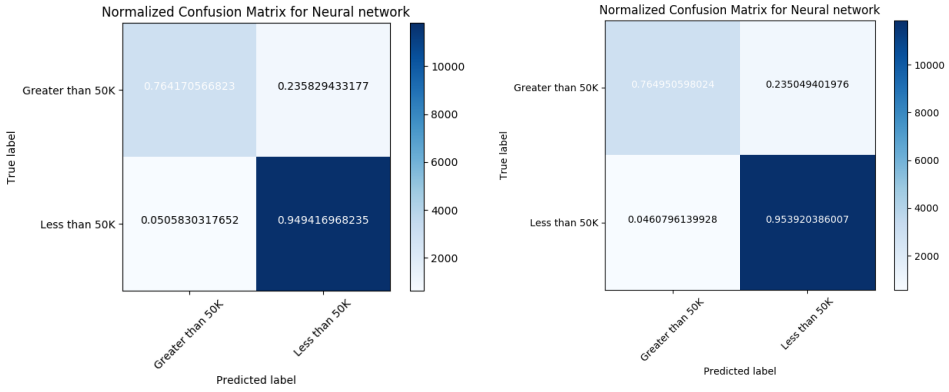


Fig 7. Normalized confusion matrix for parameter sets for MLPC obtains best precision(left) and recall(right)

The output for final classification based on best parameter set for precision and recall is:

Precision	Recall
MLPC accuracy: 0.9056569006817763, MLPC precision: 0.8761832345443508, MLPC recall: 0.8567937675287469, MLPC Fscore: 0.8658729852588029	MLPC accuracy: 0.9092807567102758, MLPC precision: 0.8830856277420753, MLPC recall: 0.8594354920155793, MLPC Fscore: 0.8703695646599223

4.2.4 Bagging Decision Tree (B-DT)

Bagging method is actually an ensembled algorithms, it builds several instances of a black-box estimator on random subsets of input training set and combines their individual predictions to form a final prediction. The baseline algorithm for bagging in this case is decision tree which is an easy way to learn both linear and non-linear relations from training set.

The parameter fine tune for bagging decision tree used the same technique as SGD parameter fine tune. Two parameters and totally 4 parameter sets were chosen in fine tune. Fig 8 shows the result of fine tune for precision and recall.

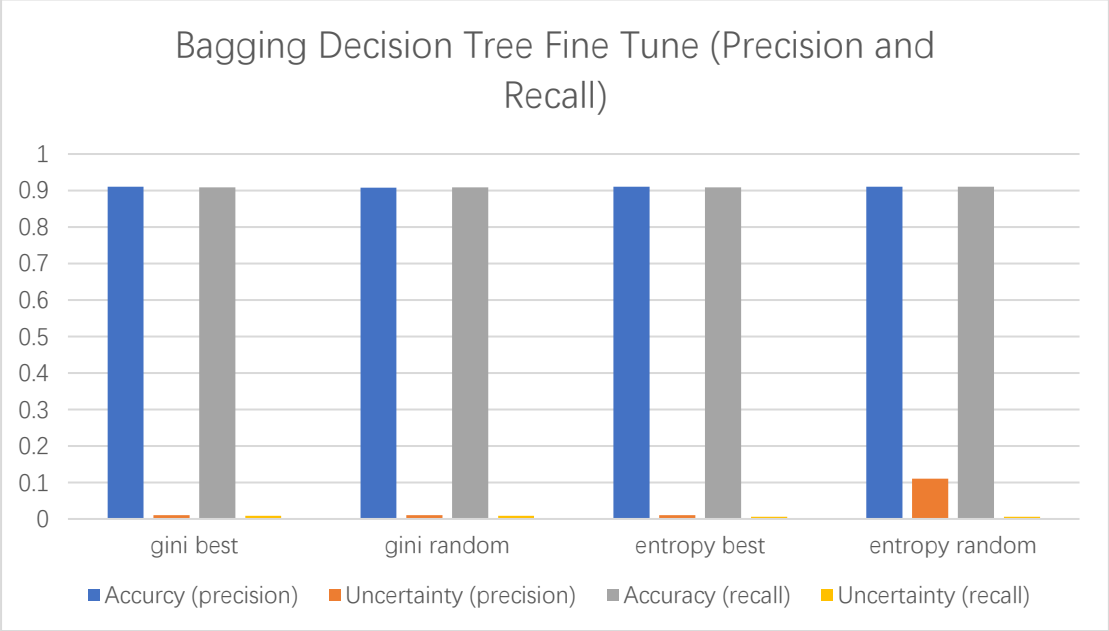


Fig 8. Bagging Decision Tree Fine Tune (Precision and Recall)

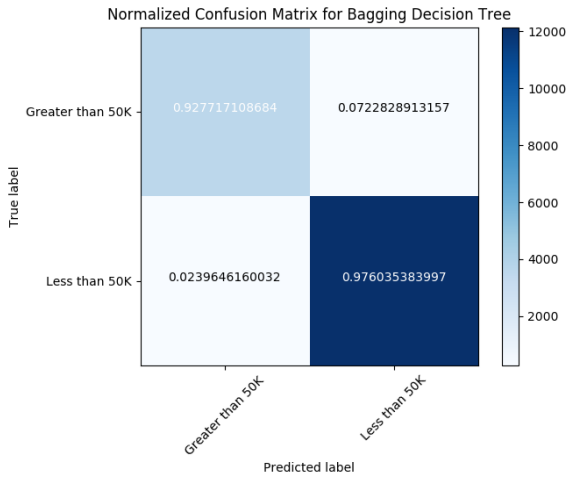


Fig 9. Normalized confusion matrix for Bagging Decision Tree

Fig 9 shown the result of result of normalized confusion matrix using parameter set obtained from fine tune progress. And the result:

Bagging Decision Tree accuracy: 0.9646213377556662,

Bagging Decision Tree precision: 0.9502627385101254,

Bagging Decision Tree recall: 0.9518762463405653,

Bagging Decision Tree Fscore: 0.9510660644364111.

5. Experiment and Discussion

5.1 Experimental result and algorithms comparison

Evaluation Metrics	Run time	10 – cross validation accuracy + error	Accuracy (over test data)	Precision (over test data)	Recall (over test data)
SVM	**24.9s**	**0.010**	**0.924**	**0.907**	**0.877**
SGD	**11.7s**	**0.007**	**0.891**	**0.915**	**0.781**
MLPC	**10.9s**	**0.012**	**0.905**	**0.876**	**0.856**
B-DT	**13.7s**	**0.018**	**0.964**	**0.950**	**0.951**

5.2 Discussion

- Discovering the relevance between features and the resulting outcomes is the key

to achieve high accuracy of classification, and quantization of such relevance will give a much cleaner dataset to classifier so as to reduce training complexity

- The most important reason that the average performance of algorithms in this project is better than the ones in some other research papers is that we also find relevance between numerical attributes and the income level. We use the extent of relevance of specific number or range of numbers of a numerical attribute as the new value for them in order to stress the impact over income, rather than staying with the original values and feeding them into classifications
- For cleaned and sorted data, complex algorithms like neural network may not perform better than some straightforward approaches, because the relations between the input features and output objectives are strong and easy to lean. In this case, a relative simple method with some enhancement to overcome its shortcomings are the perfect choice like B-DT in this project.

5.3 Personal Reflection

- Data clean and explore can reduce the complexity for training classifier, but it always need more statistical tools than what I have now.
- Basic understanding of the classification algorithms is not enough when it comes to adjust the parameters. With deeper understanding, exhaustive searching methods will be simplified by pre-selection.

6. Conclusion and Future Work

6.1 Conclusion

- Data clean and feature identification is essential for classification problem, better sorted data can improve both performance and efficiency of classifiers.
- Across algorithms the overall precision and recall are slightly different when compare with a typically part corrections such as the true positive rate for SGD is 0.571 however for MPLC this value is 0.76. For different case, different model with similar overall outcome can be considered by a particular accuracy rate such as true positive rate or false negative rate.
- Compare with the effect of data pre-processing on result accuracy the effect of parameter fine tune is relatively less.

6.2 Future Work

- Sample size in this dataset is not big enough, and there is a bias that both training and test data have more "<= 50K" samples than ">50K" ones with a ratio of 3:1. Systematic error can be raised by such bias, so some kind of normalization procedures can be applied in data pre-processing
- Bin sizes for scaling and transforming numerical attributes is important, they can be optimized by exploring data statistically
- The idea of quantizing the relevance from both categorical and numerical attributes to the objective results turn out to be a succeed in this binary classification. However, application of this idea to multi-classification tasks is much more complicated

7. Contribution

The project were separated into 5 parts, including literature review, data pre-processing, model selection, model fine tune and analysis and report writing. We worked together to do literature review shared ideas about data pre-processing and analysis techniques. After that, we decided using 4 kinds of models as mentioned previous for this project. And Xintong Chen covers data pre-processing and Qinglong Zeng covers data analysis.

	Qinglong Zeng	Xintong Chen	Work Together
Literature review			√
Data pre-processing		√	
Model selection			√
Fine Tune	√		
Report writing			√

Reference

[1] Ron Kohavi, "Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid", Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, 1996

[2] Chockalingam, V., Shah, S., & Shaw, R. *Income Classification using Adult Census Data.*

[3] Nham, T. *Classifying Income from 1994 Census Data.*

Appendix

Instruction on code:

- There are two folders for codes, namely, “Test” and “Parameter adjustment”.
- “Parameter adjustment” folder includes 4 python files for the four algorithms used in this project, they are used for adjusting parameters for each algorithm before applied on test data. The file name indicates the name of a corresponding algorithm.
- “Test” folder includes 4 python files for the four algorithms, they are used for classifying test data. The file name indicates the name of a corresponding algorithm.
- The structure of the total 8 python files are the same
- At about line 105 in each python file, there is a variable “filepath” which stores the path of train data for loading.
- At about line 140 in each python file, there is a variable “filepath_test” which stores the path of test data for loading. Feel free to manually change test data sources to prove the accuracy and facticity of our work
- The main classifier is implemented by Dell Computer XPS-13 with Inter Core i-5 CPU, the running time of each algorithms has specified in report.