Final Project: Machine Learning Challenges

Final Report

AI 539

Shengxuan Wang

wangshe@oregonstate.edu

Problem and Beneficiaries

This project is trying to solve the problem "How can we use a machine learning model to predict a people's annual income can exceed $50K?" By knowing this information, which is part of profile of people, can programs know more about their users. Furthermore, the program can predict people's behavior more correctly. Thus, the beneficiaries are companies who are using the recommendation systems to target advertising better. Because know their costumers more, means can recommend things they more likely want. For example, some E-Commerce companies.

Data Set Properties

To solve this problem, the data set "Adult", which is based on the census data, from UCI Machine Learning Repository is used in this project. The source URL of this dataset is:
https://archive-beta.ics.uci.edu/ml/datasets/adult

Data Profile:
This data has already been split into training set and test set by the source. (Notice that in the later experiment I won't use this data split, I will re-split them).

The shape of them is:

	Training set	Test Set
Number of Examples	32561	16281
Number of Features	15	15

Combine training set and test set together to look at the data profile.

The predictable class, which is counted as a feature above, is called "class", and I changed it into "income" to make it more intuitive. There are two classes of "income", ">50K" and "<=50K", which are discrete values, their distribution is:

	<=50K	>50K
Numbers	37155	11687
Percentage	76%	24%

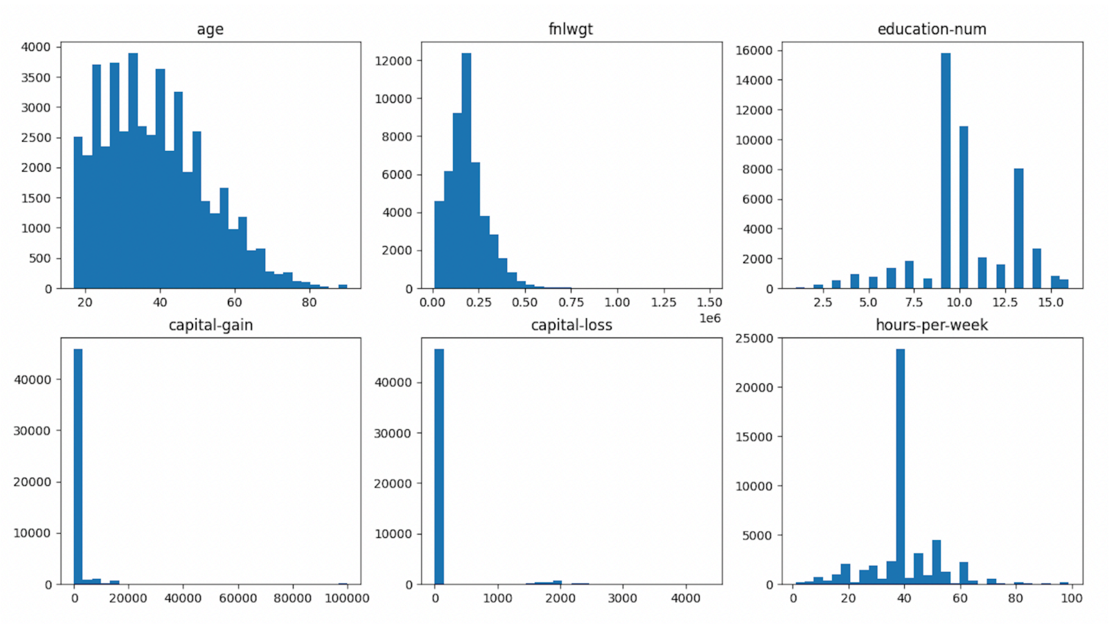
For other features, there are also continuous and discrete. I will introduce their profile separately. Start from **continuous** features.

a) Continuous Features:

Name	Type	min	max	mean	median
age	int64	17	90	38.581647	37
fnlwgt	int64	12285	1484705	189778.4	178356
education-num	int64	1	16	10.080679	10
capital-gain	int64	0	99999	1077.648844	0
capital-loss	int64	0	4356	87.303830	0
hours-per-week	int64	1	99	40.437456	40

**fnlwgt stands for final weight in CPS (Current Population Survey), means how many people the census believes that this row can represent.

In order to know more about the distribution of the data, I plot the histogram of these 6 continuous features.



b) Discrete Features

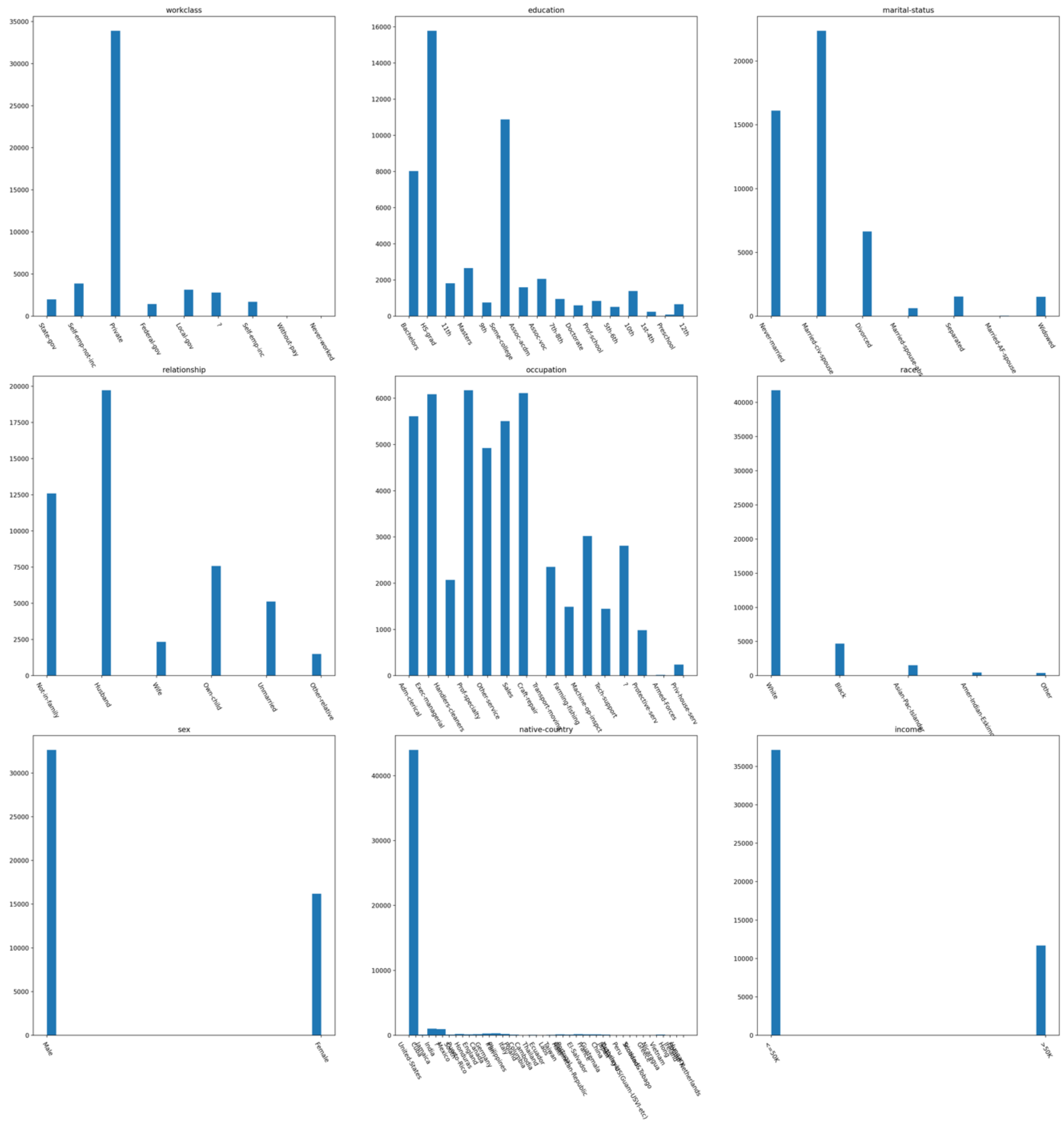
The data type of all the discrete features is 'str'.The distribution of them is shown as the table below.

Feature (Class Number)	Class	Counts
workclass	Private	33906
	Self-emp-not-inc	3862
	Local-gov	3136
	Missing values	2799
	State-gov	1981
	Self-emp-inc	1695

	Federal-gov	1432
	Without-pay	21
	Never-worked	10
education	HS-grad	15784
	Some-college	10878
	Bachelors	8025
	Masters	2657
	Assoc-voc	2061
	11th	1812
	Assoc-acdm	1601
	10th	1389
	7th-8th	955
	Prof-school	834
	9th	756
	12th	657
	Doctorate	594
	5th-6th	509
	1st-4th	247
	Preschool	83
marital-status	Married-civ-spouse	22379
	Never-married	16117
	Divorced	6633
	Separated	1530
	Widowed	1518
	Married-spouse-absent	628
	Married-AF-spouse	37
relationship	Husband	19716
	Not-in-family	12583
	Own-child	7581
	Unmarried	5125
	Wife	2331
	Other-relative	1506
occupation	Prof-specialty	6172
	Craft-repair	6112
	Exec-managerial	6086
	Adm-clerical	5611
	Sales	5504
	Other-service	4923
	Machine-op-inspct	3022
	Missing Values	2809
	Transport-moving	2355
	Handlers-cleaners	2072

	Farming-fishing	1490
	Tech-support	1446
	Protective-serv	983
	Priv-house-serv	242
	Armed-Forces	15
race	White	41762
	Black	4685
	Asian-Pac-Islander	1519
	Amer-Indian-Eskimo	470
	Other	406
sex	Male	32650
	Female	16192
native-country **because there are too many classes in this feature, only show the first 15 countries with an "other", full counting list can be found in the appendix a)	United-States	43832
	Mexico	951
	Missing Value	857
	Philippines	295
	Germany	206
	Puerto-Rico	184
	Canada	182
	El-Salvador	155
	India	151
	Cuba	138
	England	127
	China	122
	South	115
	Jamaica	106
	Italy	105
	Others	1316
income	<=50K	37155
	>50K	11687

The histogram of these 9 discrete features to look the distribution of them is shown below (If it is not clear enough, full resolution figure can be found in the appendix b)):



Data Example:

a)

age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	income
52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	45	United-States	>50K

b)

age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	income
54	?	180211	Some-college	10	Married-civ-spouse	?	Husband	Asian-Pac-Islander	Male	0	0	60	South	>50K

We can see that in this example, there are two missing values in "workclass" and "occupation".

c)

age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	income
25	Private	289980	HS-grad	9	Never-married	Handlers-cleaners	Not-in-family	White	Male	0	0	35	United-States	<=50K

Machine Learning Models

This project will use:
a) **Random Forest Classifier**: random forest can performance a very good result without too many data preprocessing, using it could process a better model more easily. But if there are too many noises in the data, overfitting will occur.
b) **Logistic Regression Classifier**: logistic regression is a simple classifier, there will be more space for improvement, easier to see the improvement when apply some challenge solution or hyperparameter choice. But logistic regression cannot handle the dependent features well.

Hyperparameter:
For all the classifier, set random state to 22.
Random Forest Classifier: try the number of trees in the forest (*n_estimators*) are 10, 20, 30, 40, and 50.
Logistic Regression Classifier: try the norm method of the penalty (*penalty*) are no penalty, L1 penalty and L2 penalty; use default solver (*solver*) "lbfgs", but change it to "liblinear" when try L1 penalty.

Hyperparameters experiment will be shown in the Experiment and Results section. Other unmentioned hyperparameters will use the default setting. Hyperparameter may change in the later experiment.

Evaluation

Metrics:
I will use accuracy as the main metrics to evaluate all the models. I will also use precision and recall when trying to solve the class imbalance problem.

Data split:
The source already provides us training data set and test data set. But, in order to create more randomness, I will mix all the data together, then randomly split 70% of the whole data set as new training set, the 15% of it as validation set, and the rest 15% of it as new test set. (Keep the random state fixed to make each run has the same result.) The training set, validation set, and test set which is going to be mentioned below all means the new split data set.

Baseline:
In this project, because our main purpose is to solve the data challenge, instead of using dummy classifier as baseline, I will use each classifier without any challenge solution as the baseline. For the missing value, I delete all the examples with missing value, and let the examples with missing value abstain in the validation/test set. (The baseline choice is different from the original idea, which will be explained in the section reflection.)

Challenges

I found there are three challenges in this data, missing values, sampling bias, and class imbalance.
Missing Values:
There are three features which are containing missing values, and their amounts are in the parentheses: ['workclass' (1836), 'occupation'(1843), 'native-country' (583)], which are shown as "?".

Sampling Bias (on Capital-gain and Capital-loss):
In the feature Capital-gain and Capital-loss, most people have no gain or loss, only a few people have some Capital-gain or Capital-loss.

Class imbalance (on income):
There is only 24% people have more than 50K income, but 76% people have less than 50K income.

Solutions

Missing Values:
(1) After I plot the histogram with class of these three features (figures on Appendix c)), I found some bias pattern on them: the missing values are always happened in the class

<= 50K, the distribution is not even but bias. So, I want to **drop all the examples with missing value to train the module**, then use the majority to predict the example with missing value, **always predict <=50K when there is missing value**.
(2) Impute the missing values using the **mode value** of the feature.
(3) Delete all the **features** with missing value.
(4) Especially in the feature “native-country”, impute the missing values with “unknown”, **remain the property that we don’t know their native country**.

Sampling Bias (on Capital-gain and Capital-loss):
(1) Maybe some people don’t want to reveal their capital income so might as well fill a “0”. This feature already cannot express true situation. So just **delete these two features**.
(2) Maybe because it is true that there are rare people having capital expenditure. **Remain** this distribution. But **re-scale** the data: e.g., **binary** them to “have or don’t have”.
(3) Using majority (figures on Appendix d)) to predict the class: e.g., if an example has capital gain or capital loss, predict it as > 50K

Class Imbalance (on income):
(1) **Undersample** the majority class.
(2) **Augment** more >50K data, by doubling the existed >50K examples.
(3) Add **class weight** on minority class.

Experiment and Results

From the data profile section, we know that there are discrete values in some features, whose data type is string. Therefore, in order to feed the data into the module, I need encode these string data. In this experiment, I use the one hot encoding method to do that.

As I mentioned in evaluation section, I merge the original training data and test data together, then randomly split the whole set, 70% as training set, 15% as validation set, 15% as test set. In this way we can get more randomness. In order to guarantee each repetition is using the same data, remain random state as 1.

Examine all the hyperparameters of classifiers and get baseline
Delete all the examples with missing value in the training set and use it to train the model, and test it on the validation set, let the examples with missing value in the validation set abstain. Here are the accuracies of the classifiers and their hyperparameters.

Random Forest:

NUM_TREE	10	20	30	40	50
Accuracy	0.7802	0.7851	0.7875	0.7890	0.7873

Logistic Regression:

PENALTY	'none'	'l2'	'l1'
SOLVER	'lbfgs'	'lbfgs'	'liblinear'
Accuracy	0.7375	0.7383	0.7849

Highlighted he best performance, pick the classifier with these hyperparameter as the **baseline**.

Challenge 1: Missing value

Solution 1: Drop all the examples with missing value to train, and always predict <=50K (majority) when there is missing value in validation set.

Random Forest	Logistic Regression
0.8452	0.8411

Solution 2: Impute the missing values using the mode value of the feature, preventing the information leakage, only use the mode from training set.

Random Forest	Logistic Regression
0.8544	0.8564

Solution 3: Delete all the feature with missing value.

Random Forest	Logistic Regression
0.8459	0.8467

Solution 4: Remain the property of all the missing values that we don't know them. Which means see "missing value" as a value.

Random Forest	Logistic Regression
0.8556	0.8568

Conclusion:

	Baseline	Solution 1	Solution 2	Solution 3	Solution 4
Random Forest	0.7890	0.8452	0.8544	0.8459	0.8556
Logistic Regression	0.7849	0.8411	0.8564	0.8467	0.8568

All the results are better than the baseline. For both classifiers, solution 4 has the best result, so remain the data processing in solution 4, move on to challenge 2.

Challenge 2: Sampling bias (on Capital-gain and Capital-loss)
Solution 1: Delete these two features.

Random Forest	Logistic Regression
0.8221	0.8361

Solution 2: Remain this distribution. But re-scale the data: e.g., binary them to "have or don't have".

Random Forest	Logistic Regression
0.8344	0.8493

Solution 3: Using majority to predict the class: e.g., if an example has capital gain or capital loss, predict it as > 50K.

Random Forest	Logistic Regression
0.8126	0.8194

Conclusion:

	Baseline	Solution 1	Solution 2	Solution 3
Random Forest	0.7890	0.8221	0.8344	0.8126
Logistic Regression	0.7849	0.8361	0.8493	0.8194

All the results are better than the baseline. For both classifiers, solution 2 has the best result, but the result accuracy is even lower than after solving the challenge 1. Therefore, back to the data processing in solution 4 of challenge 1, move on to the challenge 3.

Challenge 3: Class imbalance (on income)
Solution 1: Undersample the majority class, only use half of the majority class examples.

Random Forest	Logistic Regression
0.8082	0.8108

Solution 2: Augment more >50K data.

Random Forest	Logistic Regression
0.8128	0.8109

Solution 3: Add class weight on minority class, make the weight of minor class is twice

bigger than the weight of major class.

Random Forest	Logistic Regression
0.8204	0.8080

Conclusion:

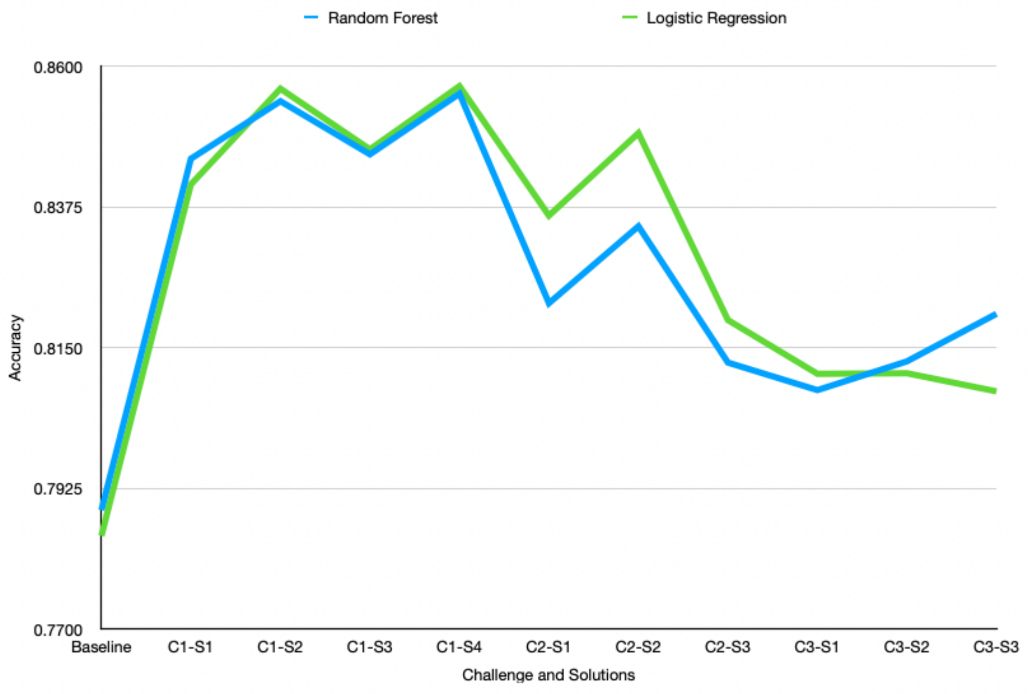
	Baseline	Solution 1	Solution 2	Solution 3
Random Forest	0.7890	0.8082	0.8128	0.8204
Logistic Regression	0.7849	0.8108	0.8109	0.8080

All the results are better than the baseline. For Random Forest, the solution 3 has the best performance, for Logistic Regression, the solution 2 has the best performance. However, they are all worse than after solving the challenge 1. So, I will keep neither solution.

Discussion:

Here is the table and graph of the overall accuracies.

	Base-line	Missing value				Sampling bias			Class imbalance		
		S1	S2	S3	S4	S1	S2	S3	S1	S2	S3
Random Forest	0.7890	0.8452	0.8544	0.8459	0.8556	0.8221	0.8344	0.8126	0.8082	0.8128	0.8204
Logistic Regression	0.7849	0.8411	0.8564	0.8467	0.8568	0.8361	0.8493	0.8194	0.8108	0.8109	0.8080



From the result, we can see that for challenge 1, solutions 4 reach the highest accuracy, then I keep this solution to experiment other solutions of the rest challenges, but all the accuracy of all solutions are worse than solution 4 for challenge 1. Therefore, solution 4 for this challenge is the most effective one. I will keep the solution 4 for challenge 1 as my final classifier method. In order to preventing information leakage, test the final classifier and baseline on the test set now.

	Baseline	Final Classifier	Improvement
Random Forest	0.7866	0.8530	14.7%
Logistic Regression	0.7828	0.8531	14.7%

Comparing to the baseline, both classifiers improve 14.7% accuracy.

Reflection:

Surprising Result

Although all the solutions I have tried made the performance better than the baseline, the most surprise part is that after I solve the challenge 1 by solution 4, all the solutions for other challenges later I have tried did not make the classifiers better. The reason why we have this kind of result may be either a) I didn't investigate enough solution for the challenges, or b) the challenge 2 and challenge 3 are not problems, for example, class imbalance is indeed a property of the data set, so maybe remaining this property can make the classifier more general, so it should not be solved.

Peer Feedback

One of my classmates is curious about why in some situation, the performance of random forest is better than that of logistic regression, in some other situations, it is reversed. Although I didn't know the answer by investigating more, my conjecture is different classifiers match different kinds of data processing, so in different data processing, the performance of these two classifiers shows different results.

Reusable Work

I have made a help file in this project, which contains some useful functions, I can reuse this file to import them and use the functions in it directly in the future work to save some time.

Change from Original Plan

In my original plan, I want to use "do nothing" as my baseline, but later, I found it is not really "do nothing". Because I used one-hot-encoder to encode the discrete value in the data set, so the encoder sees all the missing value as an individual value to encode them, so in fact, it is not "do nothing", instead, I remain the property of there is a "missing value" as a value in this feature, which is similar to one of my solutions. Eventually, I have to change my baseline to delete all the examples with missing values and let the examples with missing value in validation set abstain.

Unsolved Questions

If I have more time, I want to investigate more solutions for the challenge 2 and 3. In order to answer my question in the Surprising Result section if that is because reason a) or b). Besides, I want to try different encoder method to deal with the discrete value to see the different, and what if I also encode the continuous value? Finally, I want to answer the question in the Peer Feedback section by investigating deeply about these two classifiers.

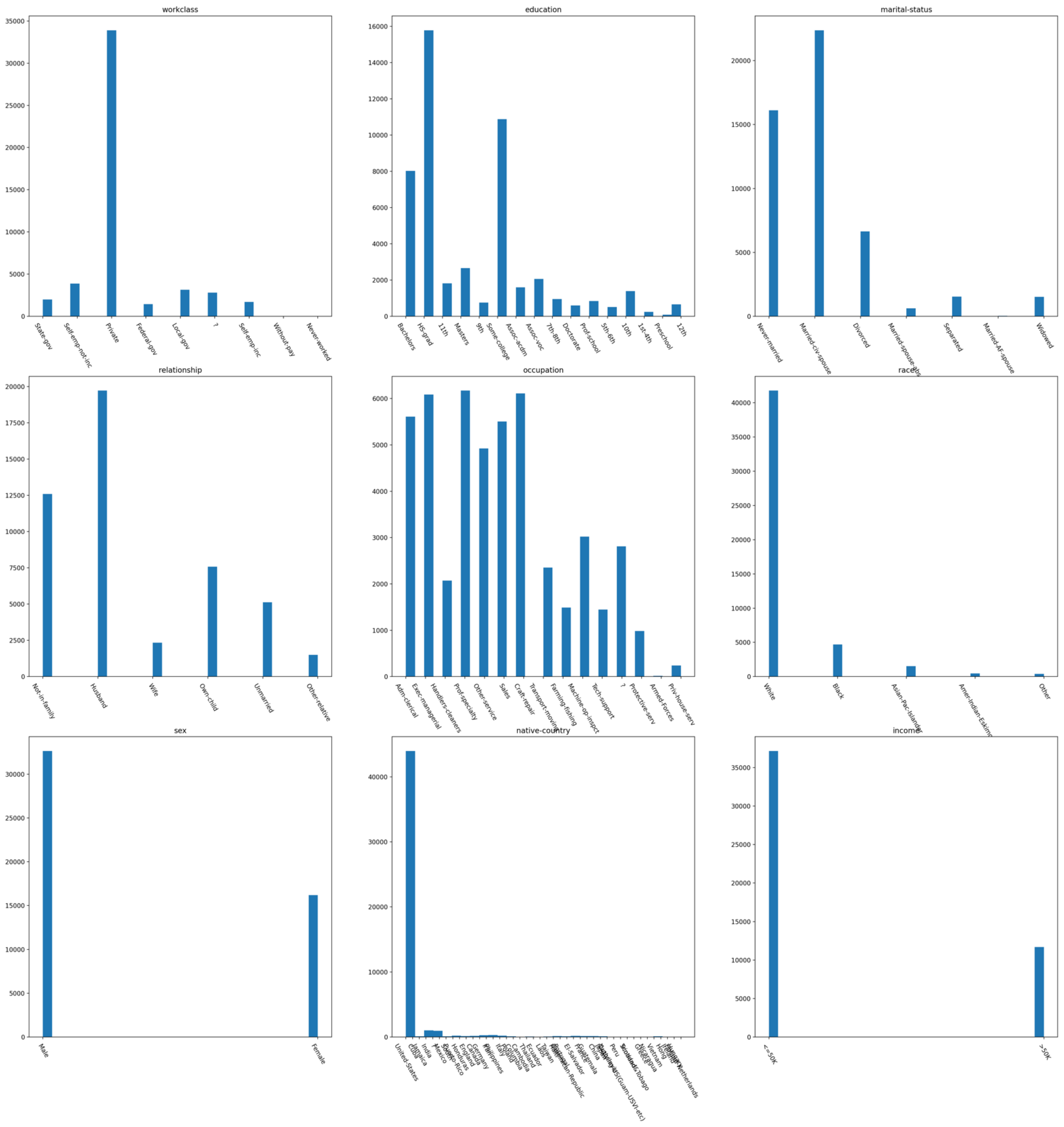
Appendix

a) Full list of the “native-country” feature:

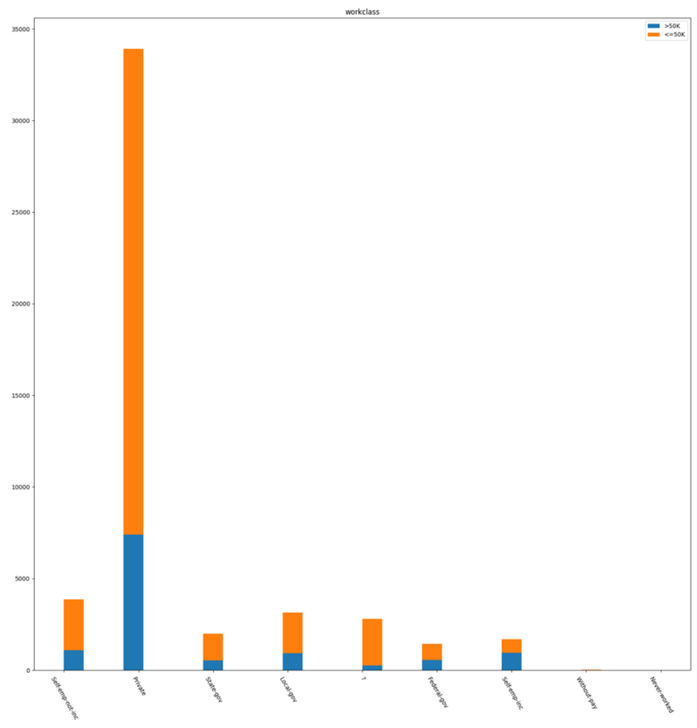
United-States	43832
Mexico	951
?	857
Philippines	295
Germany	206
Puerto-Rico	184
Canada	182
El-Salvador	155
India	151
Cuba	138
England	127
China	122
South	115
Jamaica	106
Italy	105
Dominican-Republic	103
Japan	92
Guatemala	88
Poland	87
Vietnam	86
Columbia	85
Haiti	75
Portugal	67
Taiwan	65
Iran	59
Greece	49
Nicaragua	49
Peru	46
Ecuador	45
France	38
Ireland	37
Hong	30
Thailand	30
Cambodia	28
Trinadad&Tobago	27
Laos	23
Yugoslavia	23
Outlying-US(Guam-USVI-etc)	23

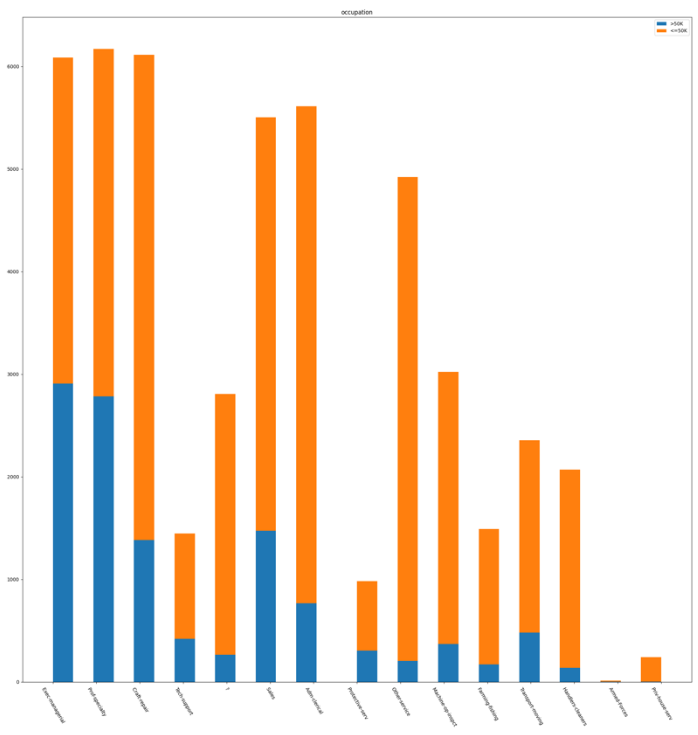
Scotland	21
Honduras	20
Hungary	19
Holand-Netherlands	1

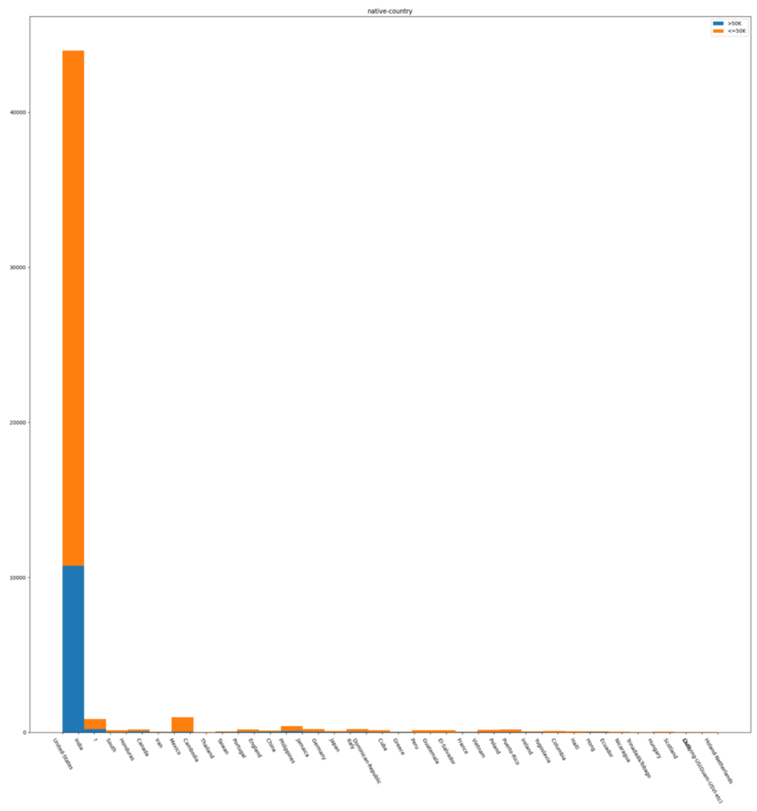
b) Full resolution histogram figure of the 9 discrete features:



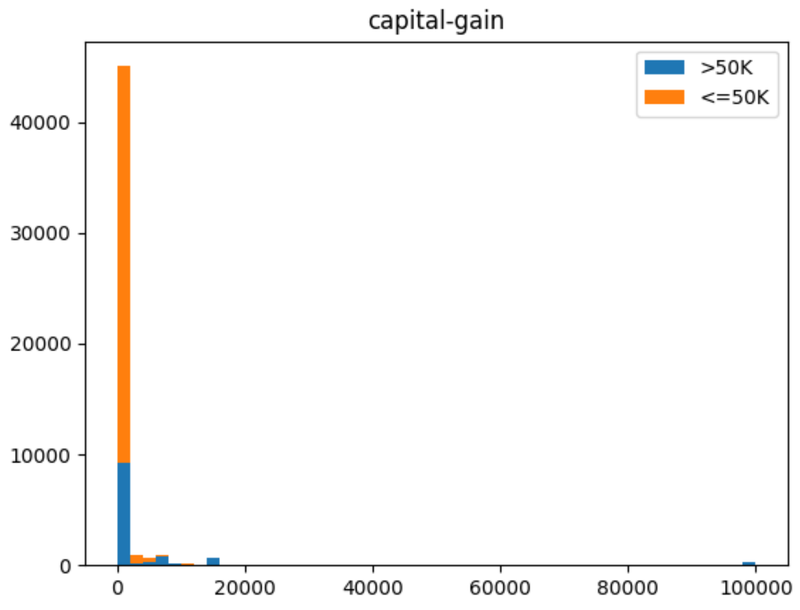
c) The histogram with class of [‘workclass’, ‘occupation’, ‘native-country’]:

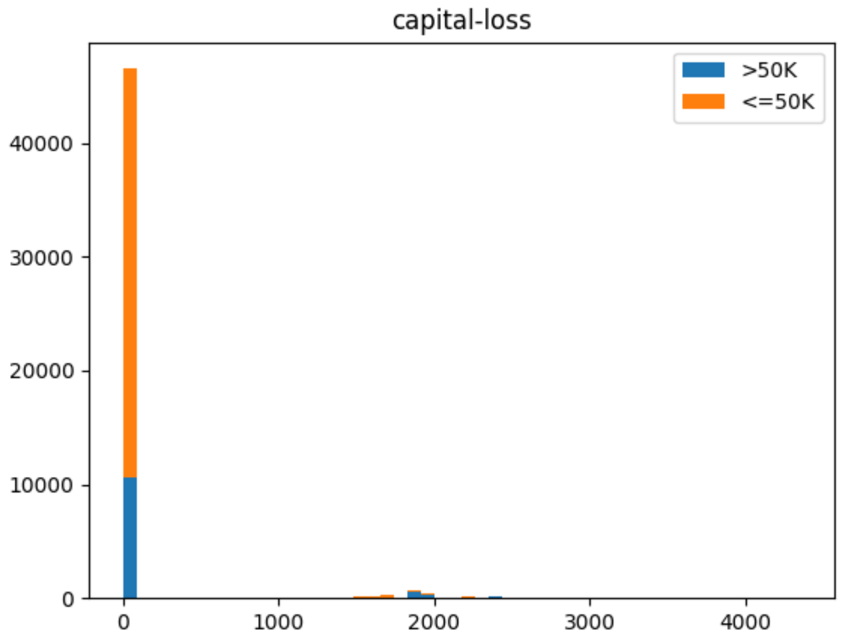






d) The histogram with class of [‘capital-gain’, ‘capital-loss]:




capital-loss
>50K
<=50K
0
10000
20000
30000
40000
0
1000
2000
3000
4000