Analysis the Income

Data Set: Adult

Yun-Te Yeh, email:yunteyeh@usc.edu

4/30/2019

1. Abstract

In order to predict a person's income, we have to analysis all features. First, in the Adult dataset (U.S Census Data), there are some important problems such as unordered categorical, missing data and so on, which have to be solved. Second, we use pandas of python to present the result. In the processing, we compare three different kinds of model such as Support Vector Machine, Naïve Bayes Classification and Random Forest. Final, choosing the best model to predict a person's income.

2. Introduction

2.1. Problem Statement and Goals

The dataset is Adult Dataset and the goal is to check whether a person's income is greater than (or less than) $50K per year. However, there are lots of problems in dataset such as numerical, ordered categorical, unordered categorical, missing data, unbalanced data and so on, so we have to solve these all problems. If we don't solve these problems, the accuracy, recall and precision are not too high, which cannot achieve our goal. Therefore, we choose the best model to train the data.

3. Approach and Implementation

3.1. Preprocessing

Before changing all features, we create the graph to investigate each feature which compares the label. As a result, we find United-States of the feature country(f14) is too big and other countries is too small in Figure.1, on the other hand, it means that United-States influences salary a lot. Thus, we can let United-States is equal 0 and other countries is equal 1. The purpose is to avoid too much features by using one-hot encoding which is from scikit-learn.

Figure.1 impact of countries on salary

Furthermore, there are three types of features such as numerical, ordered categorical and unordered categorical in adult dataset. First, for ordered categorical part, we use one-hot encoding to change unordered categorical to numerical. Second, ordered categorical is changed to ordered numerical. For example, there are two different parts in feature label, which are the bigger than 50K and the less than 50K. In order to change categorical, we let the bigger than 50K is equal 1 and the less than 50K is equal 0. As a result, the feature label becomes numerical. Third, we let the missing data (question mark) be the new feature when we use one-hot encoding. Final, we standardized all features because we can find the max value of F11 and F12 are bigger than mean in Table.1

	F1	F3	F5	F11	F12	F13
COUNT	32561.00	3.25	32561.00	32561.00	32561.00	32561.00
MEAN	38.58	1.89	10.08	1077.64	87.30	40.43
STD	13.64	1.05	2.57	7385.29	402.96	12.34
MIN	17.00	1.22	1.00	0.00	0.00	1.00
25%	28.00	1.17	9.00	0.00	0.00	40.00
50%	37.00	1.78	10.00	0.00	0.00	40.00
75%	48.00	2.37	12.00	0.00	0.00	45.00
MAX	90.00	1.48	16.00	99999.00	4356.00	99.00

Table.1 dataset information

3.2. Feature extraction and dimensionality adjustment

First, we use matplotlib.pyplot and seaborn to observe all features. From the impact of every categorical features on salary in Figure.2, we can find all categorical features affect the salary, so we cannot remove any features.

Figure.2 the impact of categorical features on salary

Second problem is that there are lots of features, when we use the one-hot encoding because all new features depends on how much categorical the features have. In order to solve this problem, we use the Principal Component Analysis (PCA) from scikit-learn to reduce the dimensionality. It can reduce all features to two dimensionalities, which means some models can use this two dimensionalities dataset such as Support Vector Machine. [1]

3.3. Compensation for unbalanced data

From the Figure.3, we can see it is unbalanced data, the less than 50K is 75.91% and the bigger than 50K is 24.08%, so there are two ways to let dataset to be balance. One is over-sampling and the other is under-sampling. We use the under-sampling way. [2]

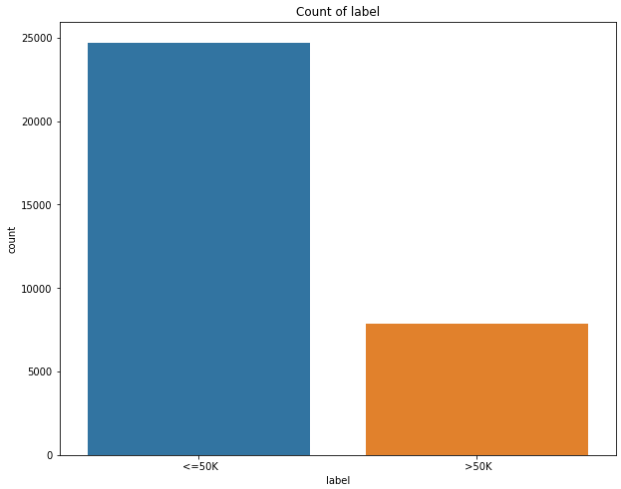

Figure. 3 count of label

3.4. Dataset Usage

My dataset is the adult_train.csv which is run the model. In the processing, we use train_test_split from scikit-learn to split two datasets and there are 60 % of training set and there are 40 % of testing set.

When we use any models, we will use cross validation to gain the best parameter. Moreover, we use the GridSearchCV from scikit-learn to find the best parameter. Comparing to the traditional method, the GridSearchCV is easier and more convenient. There are two models should use the GridSearchCV to find the parameter such as SVM and Random Forest. In the processing, both of them randomly split five datasets from training set to find the parameter. [3]

At the end, we use training set to fit model and use testing set to obtain the accuracy and confusion matrix, f1-score and so on.

3.5. Training and Classification

- Support Vector Machine (SVM)

SVM is a classifier to analyzes data points and the code is SVC from scikit-learn. The output of SVM exists the max margin between two different labels. Moreover, the hyperplane is not only linear, but also nonlinear.

In order to fit SVM model, first step, we change the ordered categorical by changing to ordered numerical and the unordered categorical by using one-hot encoding in the preprocessing. Moreover, we standardized all features. The last one is to use PCA in preprocessing. Furthermore, because of unbalance data, so we use oversampling to let two different kinds of class balance.

Second step, we split training data to testing set and training set. There are 60% of training data and 40% of testing set.

Third step, we use GridSearchCV to find the best parameter, we split five datasets from training set. In processing, if kernel is rbf, we have to find gamma and C, Moreover, if kernel is linear, we only have to find C, so the total number of runs are 36 times. As a result, the parameters of SVM are C=100, gamma=0.0001 and kernel = rbf. [3]

Final step, running the model to gain the accuracy is 78.2463%, the confusion matrix is in Table.2 and the classification report is in Table.3

	>50K	**<=50K**	**ALL**
>50K	6796	3110	9906
<=50K	1192	8678	9870
ALL	7988	11788	19776

Table.2 confusion matrix of SVM

	PRECISION	**RECALL**	**F1-SCORE**	**SUPPORT**
>50K	0.85	0.69	0.76	9906
<=50K	0.74	0.88	0.80	9870
MICRO AVG	0.78	0.78	0.78	19776
MACRO AVG	0.79	0.78	0.78	19776
WEIGHTED AVG	0.79	0.78	0.78	19776

Table.3 classification report of SVM

- Naive Bayes Classification

This classification is based on Bayes' theorem and Naïve is a conditional probability model. The code is Gaussian from scikit-learn. [4]

First step, although Naïve Bayes Classification can be used without reducing dimensionalities, the score of accuracy, recall and f1-score are lower than the model which reduces the dimensionalities. Thus, we do the same things as SVM in preprocessing.

Second step, we split training data to testing set and training set. There are 60% of training data and 40% of testing set.

Final step, running the model to gain the accuracy is 76.5625%, the confusion matrix is in Table.4 and the classification report is in Table.5

	>50K	**<=50K**	**ALL**
>50K	7056	2850	9906
<=50K	1785	8085	9870
ALL	8841	10935	19776

Table.4 confusion matrix of Naive Bayes Classification

	PRECISION	**RECALL**	**F1-SCORE**	**SUPPORT**
>50K	0.80	0.71	0.75	9906
<=50K	0.74	0.82	0.78	9870
MICRO AVG	0.77	0.77	0.77	19776
MACRO AVG	0.77	0.77	0.76	19776
WEIGHTED AVG	0.77	0.77	0.76	19776

Table.5 classification report of Naive Bayes Classification

- Random Forest

The Random Forest is an ensemble learning method for classification and is based on Decision Trees.

First step, the Random Forest model doesn't need to reduce the dimensionalities and standardized all features' values, but we have to change categorical to numerical.

Second step, we split training data to testing set and training set. There are 60% of training data and 40% of testing set.

Third step, in order to find the parameter, we have to find n_estimators, max_features, max_depth and criterion, so the total number of runs are 60

times. As a result, The parameters of Random Forest are n_estimators = 500, max_features = auto, max_depth = 8 and criterion = gini. [3] [4]

Final step, the accuracy is 83.0198%, the confusion matrix is in Table.6 and the classification report is in Table.7

	>50K	**<=50K**	**ALL**
>50K	7527	2379	9906
<=50K	979	8891	9870
ALL	8506	11270	19776

Table.6 the confusion matrix of Random Forest

	PRECISION	**RECALL**	**F1-SCORE**	**SUPPORT**
>50K	0.88	0.76	0.82	9906
<=50K	0.79	0.90	0.84	9870
MICRO AVG	0.83	0.83	0.83	19776
MACRO AVG	0.84	0.83	0.83	19776
WEIGHTED AVG	0.84	0.85	0.83	19776

Table.7 the classification report of Random Forest

4. Comparison, Results, and Interpretation

This project discusses whether a person's income is greater than (or less than) $50K per year, so our goal doesn't depend on the accuracy score. We have to see f1-score. Comparing all model in table. 8, we can find Random Forest is the best model because all of scores is better than others.

	SVM	Naïve Bayes Classification	Random Forest
accuracy	0.78	0.76	0.83
Precision	>50K:0.85 <=50K:0.74	>50K:0.80 <=50K:0.74	>50K:0.88 <=50K:0.90
recall	>50K:0.69 <=50K:0.88	>50K:0.71 <=50K:0.82	>50K:0.76 <=50K:0.84

F1-score	>50K:0.76 <=50K:0.80	>50K:0.75 <=50K:0.78	>50K:0.82 <=50K:0.84

Table.8 comparison of all models

5. Summary and conclusions

In the processing, at the first time we don't solve unbalance, we find the scores of recall, precision and f1-score are lower than model which using oversampling especially the <=50K of label, so solving unbalance is important. Moreover, in order to find relationship between features and label, we have to use graph to investigate them. If the feature doesn't relate the label, we can remove it. As a result, preprocessing plays an important role on this project whatever the model is.

References

[1] "pca_svm1," 2016. [Online]. Available: https://inclass.kaggle.com/heibankeli/pca-svm1.

[2] "porto simple EDA with python(unbalanced data)," Feb 2019. [Online]. Available: https://www.kaggle.com/lsjsj92/porto-simple-eda-with-python-unbalanced-data.

[3] "Random Forest using GridSearchCV," March 2019. [Online]. Available: https://www.kaggle.com/sociopath00/random-forest-using-gridsearchcv.

[4] "Naive Bayes Classifier in Python | Naive Bayes Algorithm | Machine Learning Algorithm | Edureka," 26 July 2017. [Online]. Available: https://www.youtube.com/watch?v=vz_xuxYS2PM.