Kaggle Project

Tianyi Chen

Chinese Academy of Sciences

2024-3-7

Overview

data preprocessing

data preprocessing

data preprocessing

data preprocessing

data preprocessing

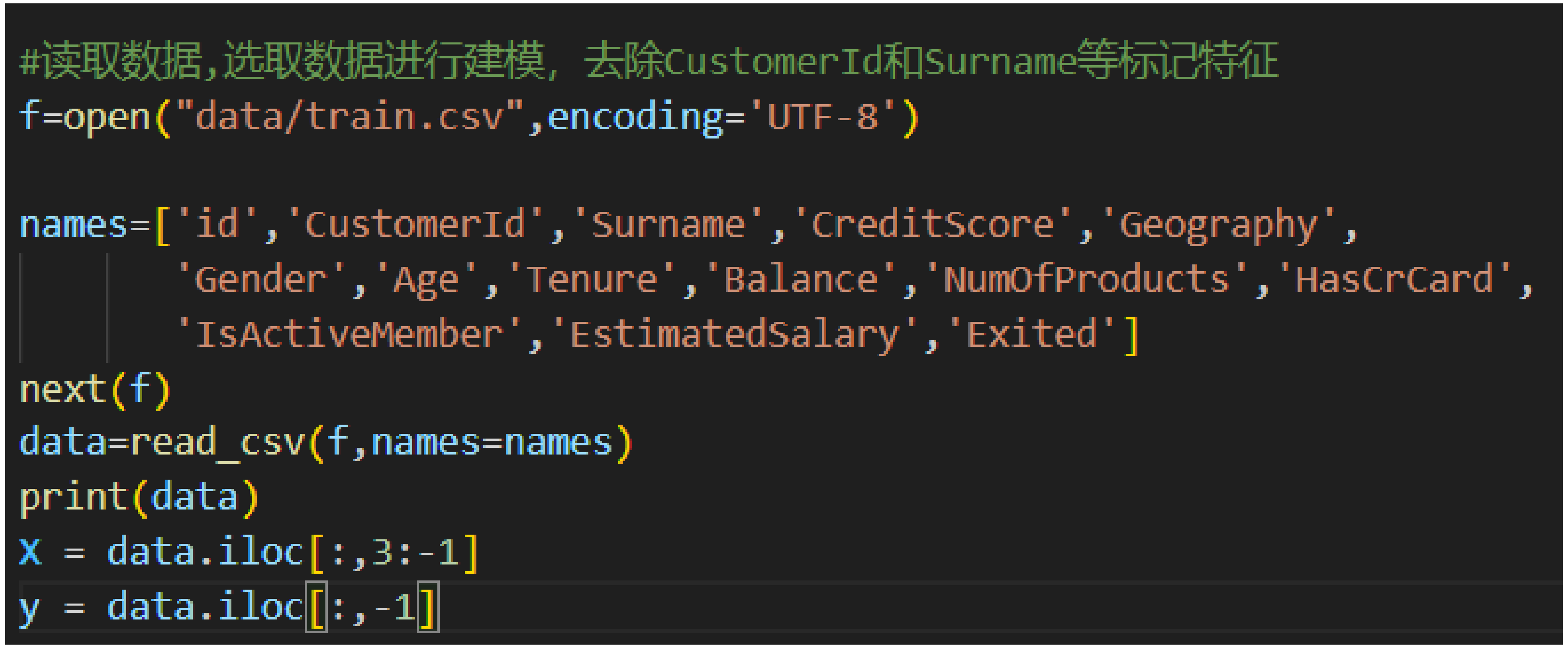
Modeling

Result

data preprocessing

data preprocessing
data preprocessing
data preprocessing
data preprocessing
data preprocessing
Modeling
Result

Import data, select data for modeling, and exclude meaningless mark data such as CustomerId and Surname



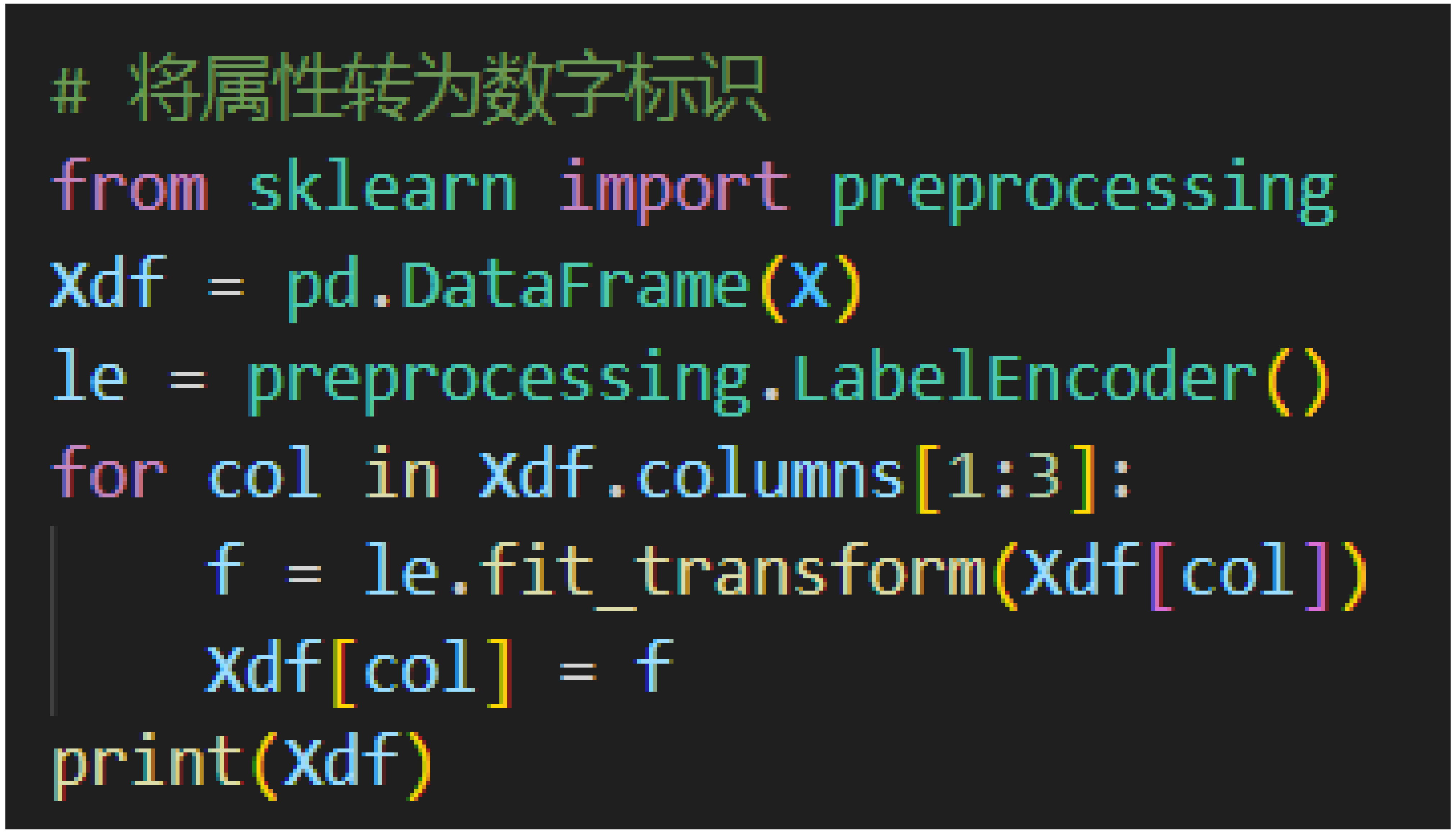
```
#读取数据,选取数据进行建模，去除CustomerId和Surname等标记特征
f=open("data/train.csv",encoding='UTF-8')

names=['id','CustomerId','Surname','CreditScore','Geography',
       'Gender','Age','Tenure','Balance','NumOfProducts','HasCrCard',
       'IsActiveMember','EstimatedSalary','Exited']
next(f)
data=read_csv(f,names=names)
print(data)
X = data.iloc[:,3:-1]
y = data.iloc[:,-1]
```

Figure 1: Import data

data preprocessing

data preprocessing
data preprocessing
data preprocessing
data preprocessing
data preprocessing
Modeling
Result

There are "Geography" and "Gender" which are text, we need to convert them into numbers



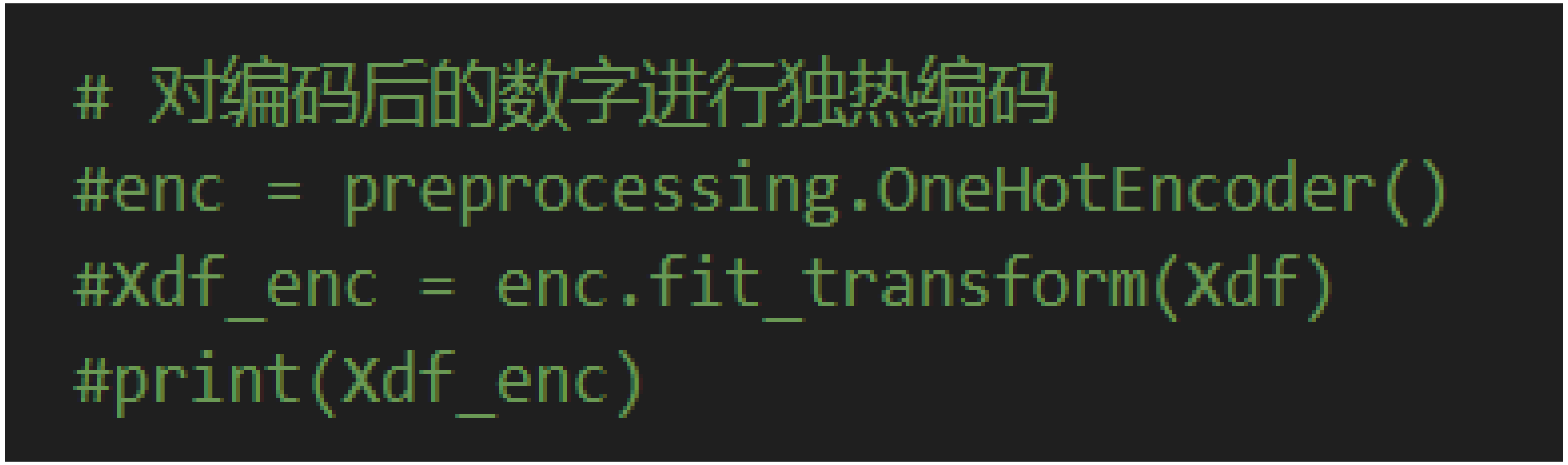
```
# 将属性转为数字标识
from sklearn import preprocessing
Xdf = pd.DataFrame(X)
le = preprocessing.LabelEncoder()
for col in Xdf.columns[1:3]:
    f = le.fit_transform(Xdf[col])
    Xdf[col] = f
print(Xdf)
```

Figure 2: Convert text data

data preprocessing

data preprocessing
data preprocessing
data preprocessing
data preprocessing
data preprocessing
Modeling
Result

May need to convert the "number" into one-hot-code, or the machine may misunderstand that the numbers have size meaning. (But I have trouble converting them because the computer is out of memory?)

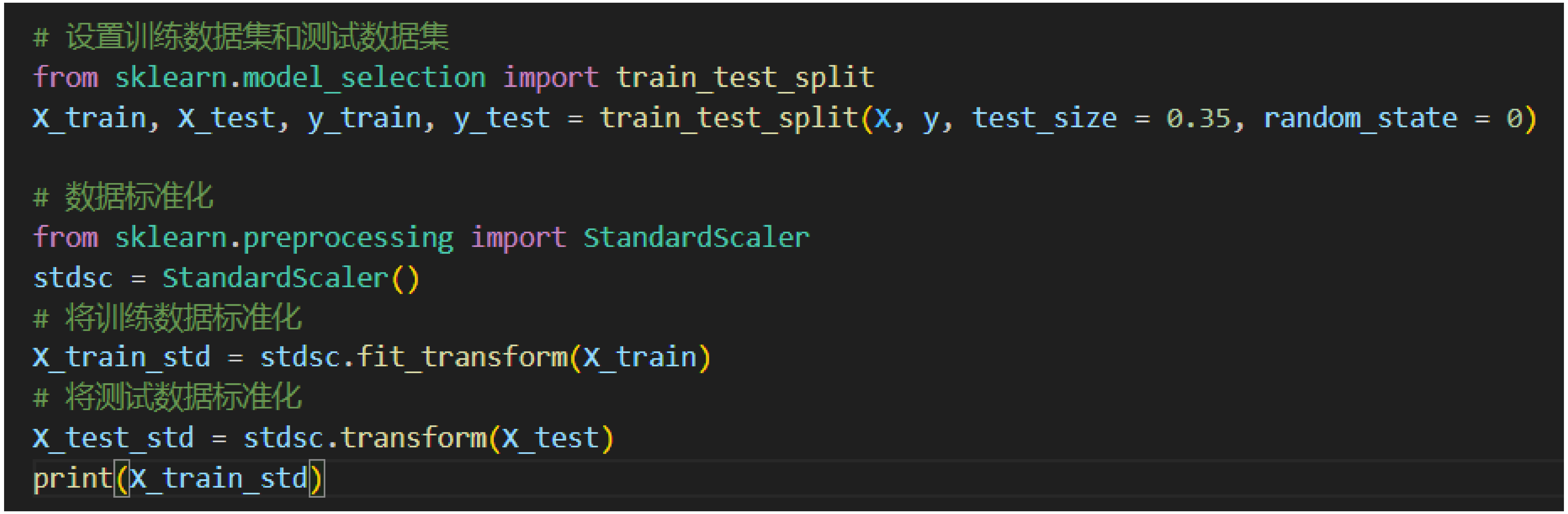


```
# 对编码后的数字进行独热编码
#enc = preprocessing.OneHotEncoder()
#Xdf_enc = enc.fit_transform(Xdf)
#print(Xdf_enc)
```

Figure 3: Trouble

data preprocessing

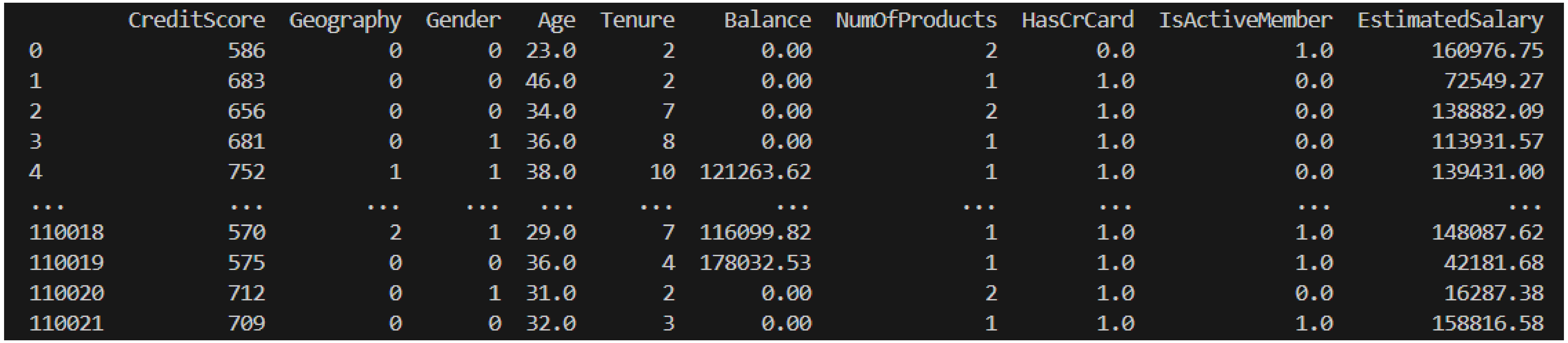
Divide the data set and standardize the data



```
# 设置训练数据集和测试数据集
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.35, random_state = 0)

# 数据标准化
from sklearn.preprocessing import StandardScaler
stdsc = StandardScaler()
# 将训练数据标准化
X_train_std = stdsc.fit_transform(X_train)
# 将测试数据标准化
X_test_std = stdsc.transform(X_test)
print(X_train_std)
```

Figure 4: Some code

data preprocessing

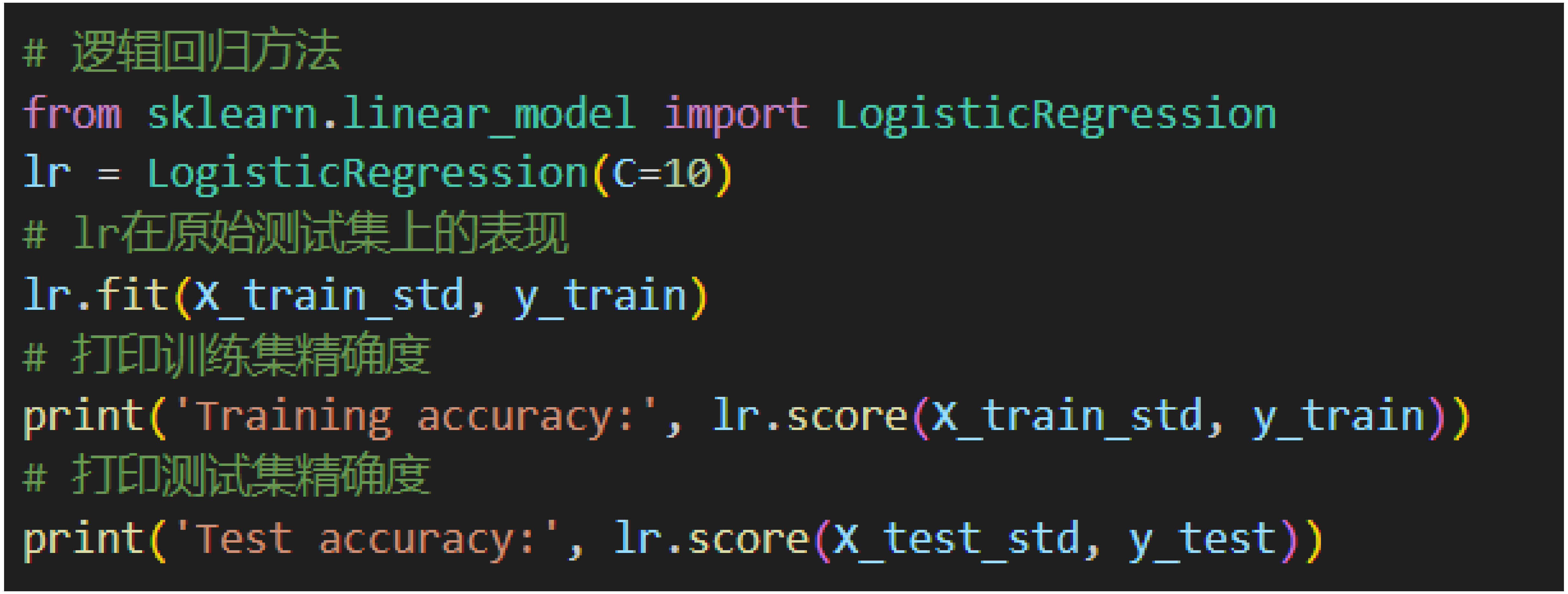


```
        CreditScore  Geography  Gender   Age  Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  EstimatedSalary
0               586          0       0  23.0       2       0.00              2        0.0             1.0        160976.75
1               683          0       0  46.0       2       0.00              1        1.0             0.0         72549.27
2               656          0       0  34.0       7       0.00              2        1.0             0.0        138882.09
3               681          0       1  36.0       8       0.00              1        1.0             0.0        113931.57
4               752          1       1  38.0      10  121263.62              1        1.0             0.0        139431.00
...             ...        ...     ...   ...     ...        ...            ...        ...             ...              ...
110018          570          2       1  29.0       7  116099.82              1        1.0             1.0        148087.62
110019          575          0       0  36.0       4  178032.53              1        1.0             1.0         42181.68
110020          712          0       1  31.0       2       0.00              2        1.0             0.0         16287.38
110021          709          0       0  32.0       3       0.00              1        1.0             1.0        158816.58
```

Figure 5: Processed Data

Modeling

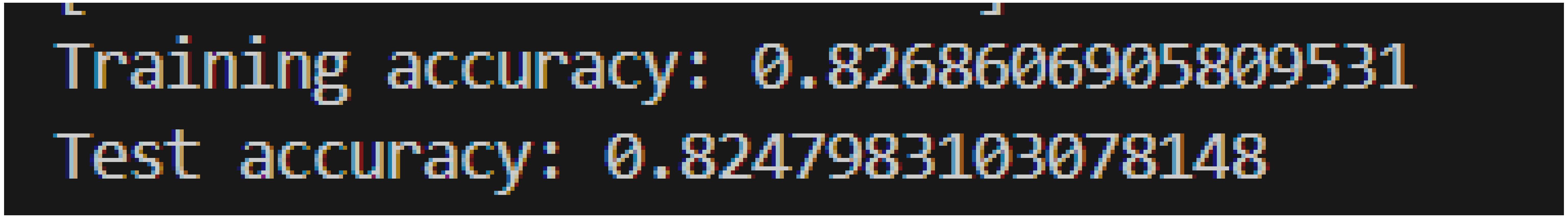
Use Logistic method



```
# 逻辑回归方法
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(C=10)
# lr在原始测试集上的表现
lr.fit(X_train_std, y_train)
# 打印训练集精确度
print('Training accuracy:', lr.score(X_train_std, y_train))
# 打印测试集精确度
print('Test accuracy:', lr.score(X_test_std, y_test))
```

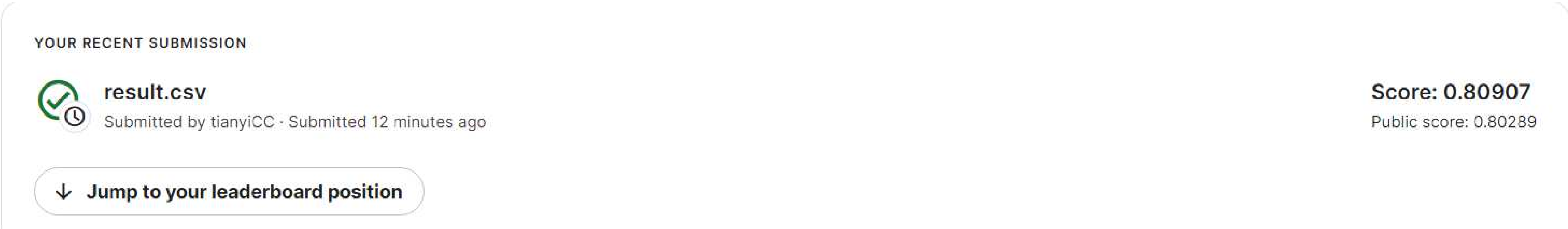
Figure 6: Some code

Result



```
Training accuracy: 0.8268606905809531
Test accuracy: 0.8247983103078148
```

Figure 7: Training result



YOUR RECENT SUBMISSION

result.csv
Submitted by tianyiCC · Submitted 12 minutes ago

Score: 0.80907
Public score: 0.80289

Figure 8: Testing result

Contact Information

Tianyi Chen
Institute of Information Engineering
Chinese Academy of Sciences

✉ chentianyi@iie.ac.cn

⌂ Team for Universal Learning and Intelligent Processing