

Project 3 - Adult Census Income

SID: 12110813

Name: 刘圣鼎

Project 3 - Adult Census Income

- 1 Problem Specification
 - 1.1 Goal
 - 1.2 Variable Name and Meaning
 - 1.3 Requirement
- 2 Data Preprocessing
 - 2.1 Read Data
 - 2.2 Data Cleaning
 - 2.3 Data Elimination
 - 2.4 Data Analysis
 - 2.4.1 Statistical Description of Numerical Variables
 - 2.4.2 Statistical Description of Discrete Variables
 - 2.4.3 Distribution Characteristics of Data
- 3 Data Modeling
 - 3.1 Data Encoding
 - 3.2 Hierarchical Cross Validation
- 4 Model Selection and Evaluation
 - 4.1 Logistic Regression
 - 4.1.1 Model Establishment and Prediction
 - 4.1.2 Model Evaluation
 - 4.2 K-Nearest-Neighbour
 - 4.2.1 Model Establishment and Prediction
 - 4.2.2 Model Evaluation
 - 4.3 Gradient-Boosting-Decision-Tree
 - 4.3.1 Model Establishment and Prediction
 - 4.3.2 Model Evaluation
 - 4.3.3 Compare with One-Hot-Encoding GBDT
 - 4.4 K-Nearest-Neighbour with GridSearchCV
 - 4.4.1 Model Establishment and Prediction
 - 4.4.2 Model Evaluation

4.5 Gradient-Boosting-Decision-Tree with GridSearchCV

4.5.1 Model Establishment and Prediction

4.5.2 Model Evaluation

5 Write Testlabel

6 Conclusion

6.1 Model Ranking

6.2 Inspiration and Thinking

1 Problem Specification

1.1 Goal

This data was extracted from the 1994 Census bureau database by Ronny Kohavi and Barry Becker (Data Mining and Visualization, Silicon Graphics). The prediction task is to determine whether a person makes over \$50K a year.

1.2 Variable Name and Meaning

- **age:** the working age of each data sample, which is a numerical variable;
- **workclass:** type of work, where there are private, local government, etc., is a character type variable;
- **fnlwgt:** the number of observational representatives of a sample in a state;
- **education:** the level of education of each sample;
- **education_num:** the schooling year of each sample;
- **marital_status:** marital status of each sample;
- **occupation:** the occupation of each sample;
- **relationship:** the family relationship of each sample;
- **race:** the race of each sample;
- **gender:** the sex of each sample;
- **capital_gain:** a capital gain is a profit that results from a disposition of a capital asset, such as stock, bond or real estate, where the amount realised on the disposition exceeds the purchase price;
- **capital_loss:** capital loss is the difference between a lower selling price and a higher purchase price, resulting in a financial loss for each sample ;
- **hours_per_week:** sample weekly working hours;
- **native_country:** the country where the sample is from;
- **income (trainable.txt):** income, where income is greater than 50K and less than or equal to 50K.

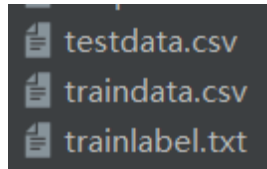
1.3 Requirement

Based on the training set, we need to train a model out of machine learning models (decision tree, nearest neighbor classifier, support vector machine, neural network as well as ensemble models and so on).

2 Data Preprocessing

2.1 Read Data

There are 3 files storing training data (traindata.csv), training label (trainlabel.txt), and test data (testdata.csv), respectively



Each line in traindata.csv corresponds to a data sample in the training dataset and different features, and each line in trainlabel.txt stores a label corresponding to the data sample in the traindata.csv. Each line in testdata.csv corresponds to a data sample.

1	age,workclass,fnlwgt,education,education.num,marital.status,occupation,relationship,race,sex,capital.gain,capital.loss,hours.per.week,native.country
2	77,Local-gov,177550,Bachelors,13,Married-civ-spouse,Adm-clerical,Husband,White,Male,0,14,United-States
3	40,Self-emp-inc,475322,Bachelors,13,Separated,Craft-repair,Own-child,White,Male,0,0,50,United-States
4	29,Self-emp-not-inc,341672,HS-grad,9,Married-spouse-absent,Transport-moving,Other-relative,Asian-Pac-Islander,Male,0,1564,50,India
5	41,Private,184378,HS-grad,9,Separated,Craft-repair,Not-in-family,White,Male,0,0,40,United-States
6	22,Private,396967,Some-college,10,Never-married,Adm-clerical,Not-in-family,White,Female,0,0,25,United-States
7	27,?,157624,HS-grad,9,Separated,?,Other-relative,White,Female,0,0,40,United-States
8	35,Private,112264,HS-grad,9,Married-civ-spouse,Sales,Husband,White,Male,0,0,50,United-States
9	48,Private,238726,HS-grad,9,Widowed,Adm-clerical,Unmarried,White,Female,0,0,40,United-States
10	37,Private,161141,HS-grad,9,Never-married,Handlers-cleaners,Not-in-family,White,Male,0,0,40,United-States
11	67,?,182378,Bachelors,13,Married-civ-spouse,?,Husband,White,Male,9386,0,60,United-States
12	26,Self-emp-not-inc,219897,Masters,14,Never-married,Prof-specialty,Not-in-family,White,Female,0,0,50,United-States
13	63,Private,101877,Assoc-acdm,12,Married-spouse-absent,Adm-clerical,Other-relative,White,Female,0,0,35,United-States
14	27,Private,85126,HS-grad,9,Never-married,Machine-op-inspct,Not-in-family,White,Female,0,0,40,United-States
15	27,Private,292120,HS-grad,9,Divorced,Tech-support,Not-in-family,White,Female,0,0,45,United-States
16	34,Private,134737,Bachelors,13,Never-married,Sales,Own-child,White,Male,0,0,40,United-States
17	33,Private,119833,9th,5,Married-civ-spouse,Handlers-cleaners,Husband,White,Male,0,0,40,United-States
18	20,Private,691830,HS-grad,9,Never-married,Sales,Own-child,Black,Female,0,0,35,United-States
19	39,Private,206528,HS-grad,9,Never-married,Adm-clerical,Not-in-family,White,Male,0,0,45,United-States
20	27,Private,203776,Assoc-acdm,12,Married-civ-spouse,Exec-managerial,Husband,White,Male,0,0,55,United-States
21	19,Private,104844,Some-college,10,Never-married,Sales,Own-child,White,Female,0,0,15,United-States
22	43,Private,122473,9th,5,Never-married,Machine-op-inspct,Unmarried,Black,Female,0,625,40,United-States
23	29,Private,271328,Bachelors,13,Never-married,Prof-specialty,Not-in-family,White,Male,4650,0,40,United-States
24	18,Private,216284,11th,7,Never-married,Adm-clerical,Own-child,White,Female,0,0,20,United-States
25	27,Private,279689,5th-6th,3,Married-civ-spouse,Farming-fishing,Husband,White,Male,0,0,40,Mexico
26	55,State-gov,296991,Prof-school,15,Married-civ-spouse,Prof-specialty,Husband,White,Male,15024,0,40,United-States
27	70,?,148065,Some-college,10,Married-civ-spouse,?,Husband,White,Male,0,0,4,United-States
28	45,Local-gov,255559,HS-grad,9,Never-married,Adm-clerical,Not-in-family,White,Female,0,0,40,United-States
29	37,Self-emp-not-inc,29814,HS-grad,9,Never-married,Farming-fishing,Unmarried,White,Male,0,0,50,United-States

Import corresponding packages in jupyter notebook, read the data from files and then preprocess the data

#导入相关库

```
import numpy as np
```

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
```

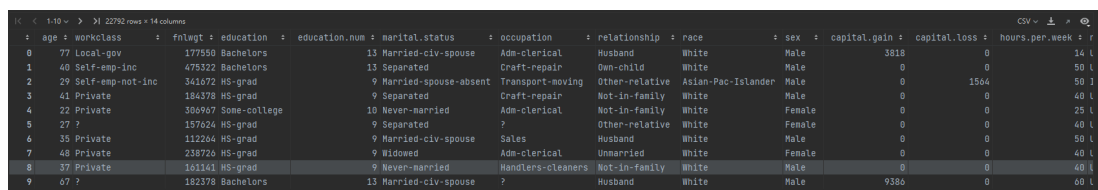
```
import seaborn as sns
```

```
%matplotlib inline
```

#读取训练集的features，读取为dataframe类型

```
df = pd.read_csv(r'.\traindata.csv')

#读取训练集的label，并将其转化为numpy数组
path="D:\PythonProject\data\trainlabel.txt"
with open(path,"r") as f:
    content=f.readlines()
target=[]
for item in content:
    target.append(int(item))
target=np.array(target)
```



	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country
0	77	Local-gov	177550	Bachelors	13	Married-civ-spouse	Adm-clerical	Husband	White	Male	3818	0	14	U.S.
1	40	Self-emp-inc	475322	Bachelors	13	Separated	Craft-repair	Own-child	White	Male	0	0	50	U.S.
2	29	Self-emp-not-inc	341672	HS-grad	9	Married-spouse-absent	Transport-moving	Other-relative	Asian-Pac-Islander	Male	0	1504	50	U.S.
3	41	Private	184378	HS-grad	9	Separated	Craft-repair	Not-in-family	White	Male	0	0	40	U.S.
4	22	Private	384907	Some-college	10	Never-married	Adm-clerical	Not-in-family	White	Female	0	0	25	U.S.
5	27	?	157624	HS-grad	9	Separated	?	Other-relative	White	Female	0	0	40	U.S.
6	35	Private	112264	HS-grad	9	Married-civ-spouse	Sales	Husband	White	Male	0	0	50	U.S.
7	48	Private	238726	HS-grad	9	Widowed	Adm-clerical	Unmarried	White	Female	0	0	40	U.S.
8	37	Private	161141	HS-grad	9	Never-married	Handlers-cleaners	Not-in-family	White	Male	0	0	40	U.S.
9	67	?	182378	Bachelors	13	Married-civ-spouse	?	Husband	White	Male	9386	0	60	U.S.

2.2 Data Cleaning

```
#查看数据缺失情况：
print(df.isnull().any())
```

```
age                False
workclass          False
fnlwgt             False
education          False
education.num      False
marital.status     False
occupation         False
relationship       False
race              False
sex               False
capital.gain       False
capital.loss       False
hours.per.week    False
native.country     False
```

From the above figure, we can find that there is no missing value.

But we can find that there exist some invalid values. I used two methods to handle the invalid character '?' here

- Use the mode to replace "?"

```
for column in df.columns:
    mode=df[column].mode()[0]
    df[column].replace('?',mode,inplace=True)
```

- Preserve "?" as a feature for subsequent one-hot-code encoding

2.3 Data Elimination

```
df[['education','education.num']].value_counts()
```

< < 1-10 > > Length: 16, dtype: int64		
education	education.num	count
HS-grad	9	7367
Some-college	10	5060
Bachelors	13	3742
Masters	14	1193
Assoc-voc	11	970
11th	7	840
Assoc-acdm	12	752
10th	6	641

From the running results of the above code, we can find that:

In the original household income dataset, there are two variables related to education level, one is education (level of education) and the other is education. num (duration of education), and the values of these two variables are one-to-one mapping, except that one is a character type and the other is a corresponding numerical type.

If both variables are included in the model, it will result in information redundancy.

The "fnlwgt" variable represents a sequence number that has no practical significance for the level of income. Therefore, in order to avoid the impact of redundant information and meaningless variables on the model, it is considered to remove the "education" variable and "fnlwgt" variable from the dataset.

```
df=df.drop(['fnlwgt','education'],axis=1)
```

2.4 Data Analysis

In the above dataset, many variables are discrete, such as education level, marital status, occupation, gender, etc. Usually, after getting the data, we need to clean it, such as checking whether there are repeated observations, missing values, Outlier, etc. In addition, if modeling, we need to recode the character-type discrete variables accordingly.

2.4.1 Statistical Description of Numerical Variables

#数值型变量统计描述

```
df.describe()
```

	age	education.num	capital.gain	capital.loss	hours.per.week
count	22792.000000	22792.000000	22792.000000	22792.000000	22792.000000
mean	38.514918	10.075509	1048.223192	85.975123	40.455291
std	13.640207	2.584257	7226.283048	400.471450	12.312654
min	17.000000	1.000000	0.000000	0.000000	1.000000
25%	28.000000	9.000000	0.000000	0.000000	40.000000
50%	37.000000	10.000000	0.000000	0.000000	40.000000
75%	47.000000	12.000000	0.000000	0.000000	45.000000
max	90.000000	16.000000	99999.000000	3900.000000	99.000000

The results described the simple statistical values of relevant numerical variables, including the number of non missing observations (count), mean, standard deviation (std), minimum (min), lower Quartile (25%), median (50%), upper Quartile (75%) and maximum (max).

2.4.2 Statistical Description of Discrete Variables

#离散型变量统计描述

```
df.describe(include= ['object'])
```

	workclass	marital.status	occupation	relationship	race	sex	native.country
count	22792	22792	22792	22792	22792	22792	22792
unique	8	7	14	6	5	2	40
top	Private	Married-civ-spouse	Prof-specialty	Husband	White	Male	United-States
freq	17157	10452	4190	9207	19476	15258	20841

The above are the statistical values of discrete variables, including the number of non missing observations for each variable (count), the number of different discrete values (unique), the highest frequency discrete value (top), and the highest frequency frequency frequency (freq).

2.4.3 Distribution Characteristics of Data

Import the pyplot and plot the distribution characteristics of data.

The following is the example of the distribution characters of numeric variables. And the code is as follows:

#以被调查居民的年龄和每周工作小时数为例，绘制各自的分布形状图：

```
import matplotlib.pyplot as plt

low=[]
high=[]
for i in range(len(target)):
    item=target[i]
    if item==0:
        low.append(i)
    else:
        high.append(i)

# 设置绘图风格
plt.style.use('ggplot')

# 设置多图形的组合
fig, axes = plt.subplots(3,1)

# 绘制不同收入水平下的年龄核密度图，观察连续型变量的分布情况
df.age[low].plot(kind = 'kde', label = '<=50K of age', ax = axes[0],
legend = True, linestyle = '-')
df.age[high].plot(kind = 'kde', label = '>50K of age', ax = axes[0],
legend = True, linestyle = '--')

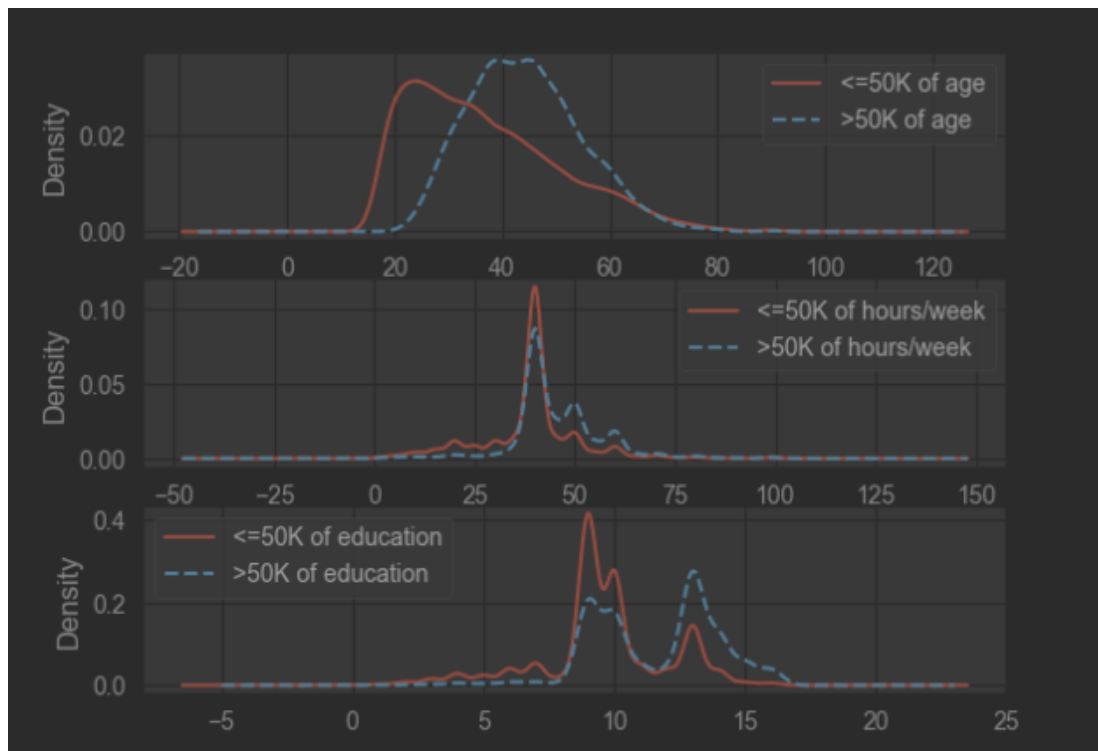
# 绘制不同收入水平下的周工作小时核密度图
df['hours.per.week'][low].plot(kind = 'kde', label = '<=50K of
hours/week', ax = axes[1], legend = True, linestyle = '-')
```



```
df['hours.per.week'][high].plot(kind = 'kde', label = '>50K of
hours/week', ax = axes[1], legend = True, linestyle = '--')
```

绘制不同收入水平下的受教育时长核密度图

```
df['education.num'][low].plot(kind = 'kde', label = '<=50K of
education', ax = axes[2], legend = True, linestyle = '-')
df['education.num'][high].plot(kind = 'kde', label = '>50K of
education', ax = axes[2], legend = True, linestyle = '--')
```



The first figure shows the nuclear density distribution of age at different income levels. For residents with an annual income of more than 50000 dollars, their age is almost normal distribution, while for residents with an income of less than 50000 dollars, their age shows the feature of right leaning. The characteristic is that there are more elderly residents than younger residents.

The second graph shows the nuclear density of weekly working hours under different income levels, and it is obvious that the distribution trends of the two are very similar, with local peaks appearing.

The third figure shows the nuclear density map of education duration under different income levels, and it is obvious that the distribution trends of the two are very similar, and there are also multiple local peaks.

3 Data Modeling

Due to the presence of many discrete variables in the dataset, whose values are strings, it is not conducive to modeling. Therefore, it is necessary to first recode these variables. There are many encoding methods:

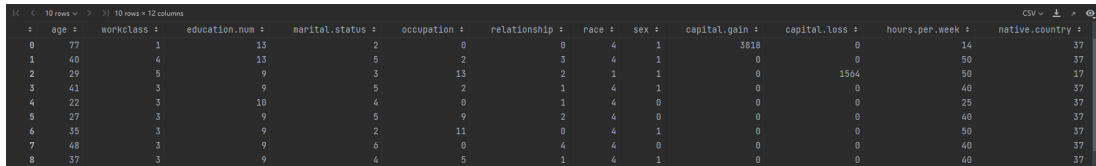
3.1 Data Encoding

In this project, I use the following two encoding methods:

- Convert character-based values to integer-based values

And the code is shown as follows:

```
# 离散型变量的重编码,将字符型的值转换为整数型的值
for feature in df.columns:
    if df[feature].dtype == 'object':
        df[feature] = pd.Categorical(df[feature]).codes
df.head(10)
```



	age	workclass	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country
0	77	1	13	2	0	0	4	1	3818	0	14	37
1	40	4	13	5	2	3	4	1	0	0	50	37
2	29	5	9	3	13	2	1	1	0	1564	50	17
3	41	3	9	5	2	1	4	1	0	0	40	37
4	22	3	10	4	0	1	4	0	0	0	25	37
5	27	3	9	5	9	2	4	0	0	0	40	37
6	35	3	9	2	11	0	4	1	0	0	50	37
7	48	3	9	0	0	4	4	0	0	0	40	37
8	37	3	9	4	5	1	4	1	0	0	40	37

- One-Hot encoding (similar to dummy variables)

When using the One-Hot encoding , preserve "?" as a feature.

And the code is shown as follows:

```

num_columns=

['age', 'fnlwgt', 'education.num', 'capital.gain', 'capital.loss', 'hours.per
r.week']

cat_columns=

['workclass', 'education', 'marital.status', 'occupation', 'relationship', '
race', 'sex', 'native.country']

#离散型变量进行热编码

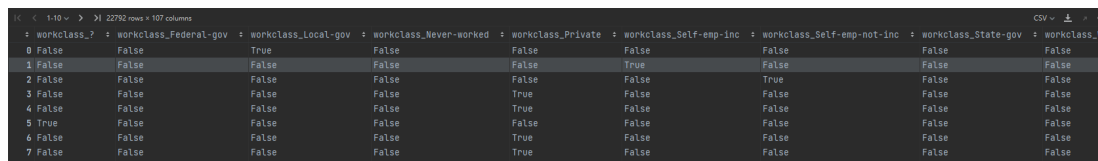
encode_df = pd.get_dummies(df, columns=cat_columns)

#连续型变量归一化

# 对连续型变量进行z-score标准化，经处理数据符合标准正态分布，均值为0，标准差
为1,然后，删除未标准化数据，将标准化后数据合并。

num_mean=encode_df[num_columns].mean()
num_std=encode_df[num_columns].std()
num_normal=(encode_df[num_columns]-num_mean)/num_std
encode_df=encode_df.drop(columns=num_columns)
encode_df=pd.concat([encode_df, num_normal], axis=1)
features=encode_df.columns
encode_df=encode_df.values

```



	workclass_Federal-gov	workclass_Local-gov	workclass_Never-worked	workclass_Private	workclass_Self-emp-inc	workclass_Self-emp-not-inc	workclass_State-gov	workclass_N
0	False	True	False	False	False	False	False	False
1	False	False	False	False	True	False	False	False
2	False	False	False	False	False	True	False	False
3	False	False	False	True	False	False	False	False
4	False	False	False	True	False	False	False	False
5	True	False	False	False	False	False	False	False
6	False	False	False	True	False	False	False	False
7	False	False	False	True	False	False	False	False
8	False	False	False	True	False	False	False	False

3.2 Hierarchical Cross Validation

Based on the "clean" dataset above, it needs to split the traindata into two parts, one for the construction of the classifier model and the other for the evaluation of the classifier model in order to avoid overfitting or underfitting of the classifier model.

- If the model performs well on the training set but poorly on the test set, it indicates that the classifier model belongs to an overfitting state.
- If the model cannot fit the data well during the training process, it indicates that the model is in an underfitting state.

Using a hierarchical cross validation approach, the entire dataset is divided into a training set and a testing set.

Usually, the proportion of training and testing sets is allocated to 75% and 25%.

The code is as follows:

```
from sklearn.model_selection import StratifiedShuffleSplit
#分层交叉验证
#使用分层交叉验证的方式，将整个数据集划分为训练集（train）和测试集（test）。
sss=StratifiedShuffleSplit(n_splits=2,train_size=0.75)
for train_index,test_index in sss.split(encode_df,target):
    trainx,testx = encode_df[train_index],encode_df[test_index]
    trainy,testy = target[train_index],target[test_index]
```

Using Stratified ShuffleSplit can ensure that the proportion of each data in the total data is basically the same before and after Stratified sampling.

4 Model Selection and Evaluation

4.1 Logistic Regression

4.1.1 Model Establishment and Prediction

Import Logistic Regression Model.

After fitting the training set data, the accuracy of the model is approximately 85.01%.

```
#导入Logistic回归模型，对训练集数据拟合后，得出模型的准确率约为85.01%。
from sklearn.linear_model import SGDClassifier
lr=SGDClassifier(loss='log',max_iter=100)
lr.fit(trainx,trainy)
print("score:",lr.score(testx,testy))
```

4.1.2 Model Evaluation

After the model is built, the next step is to use the obtained classifier to predict the test dataset and verify the model's performance outside of the sample. Usually, there are multiple methods for verifying the quality of a model.

For the continuous variables predicted, the commonly used indicators are Mean squared error (MSE) and Root-mean-square deviation (RMSE).

For the predicted Categorical variable, the commonly used indicators include the accuracy in the Confusion matrix, the area AUC under the Receiver operating characteristic, K-S value, etc.

Next, predict and evaluate the models constructed in the text in sequence.

#模型评估

```
from sklearn.metrics import  
roc_auc_score, precision_recall_curve, classification_report, roc_curve
```

#auc值和分类报告。

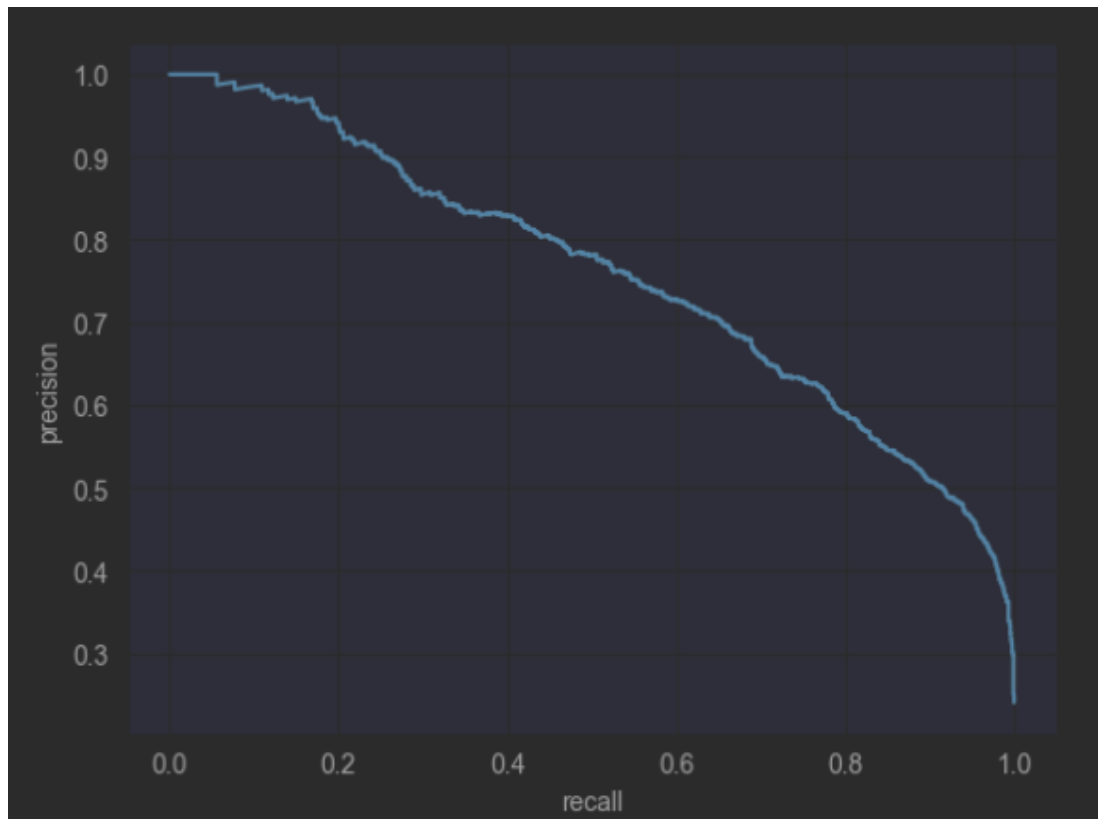
```
pred=lr.predict_log_proba(testx)[: ,1]  
pred_labels=lr.predict(testx)  
print("auc:", roc_auc_score(testy, pred))  
print(classification_report(testy, pred_labels))
```

#PR曲线

```
precision, recall, _ = precision_recall_curve(testy, pred)  
plt.plot(recall, precision)  
plt.xlabel('recall')  
plt.ylabel('precision')
```

```
score: 0.8501228501228502  
auc: 0.9033099072797337
```

	precision	recall	f1-score	support
0	0.88	0.93	0.90	4326
1	0.73	0.61	0.66	1372
accuracy			0.85	5698
macro avg	0.80	0.77	0.78	5698
weighted avg	0.84	0.85	0.85	5698



The area under the Receiver Operating Characteristic(ROC) reflects the quality of the model.

- If the area is 1, overfitting may occur;
- If it is between 0.5 and 1, it may have predictability.
- If it is less than 0.5, it is a poor model, but it can be used for reverse prediction.

In this simulation, the auc value is about 0.903, that is, the area under the Receiver operating characteristic is about 90.3%. It can be said that a relatively ideal Logistic regression model has been obtained.

4.2 K-Nearest-Neighbour

4.2.1 Model Establishment and Prediction

Firstly, for the K-nearest neighbor model, the `KNeighborsClassifier` class in the `sklearn.neighbors` is directly called here.

And the default parameters of the model are used.

That is, the K-nearest neighbor model automatically selects the best search nearest neighbor algorithm (algorithm='auto '),

Calculates the distance between samples using the Euclidean distance formula (p=2) Specify that the number of nearest neighbors for unknown classification samples is 5 (n_neighbors=5) and that all nearest neighbor samples have equal weights (weights='uniform').

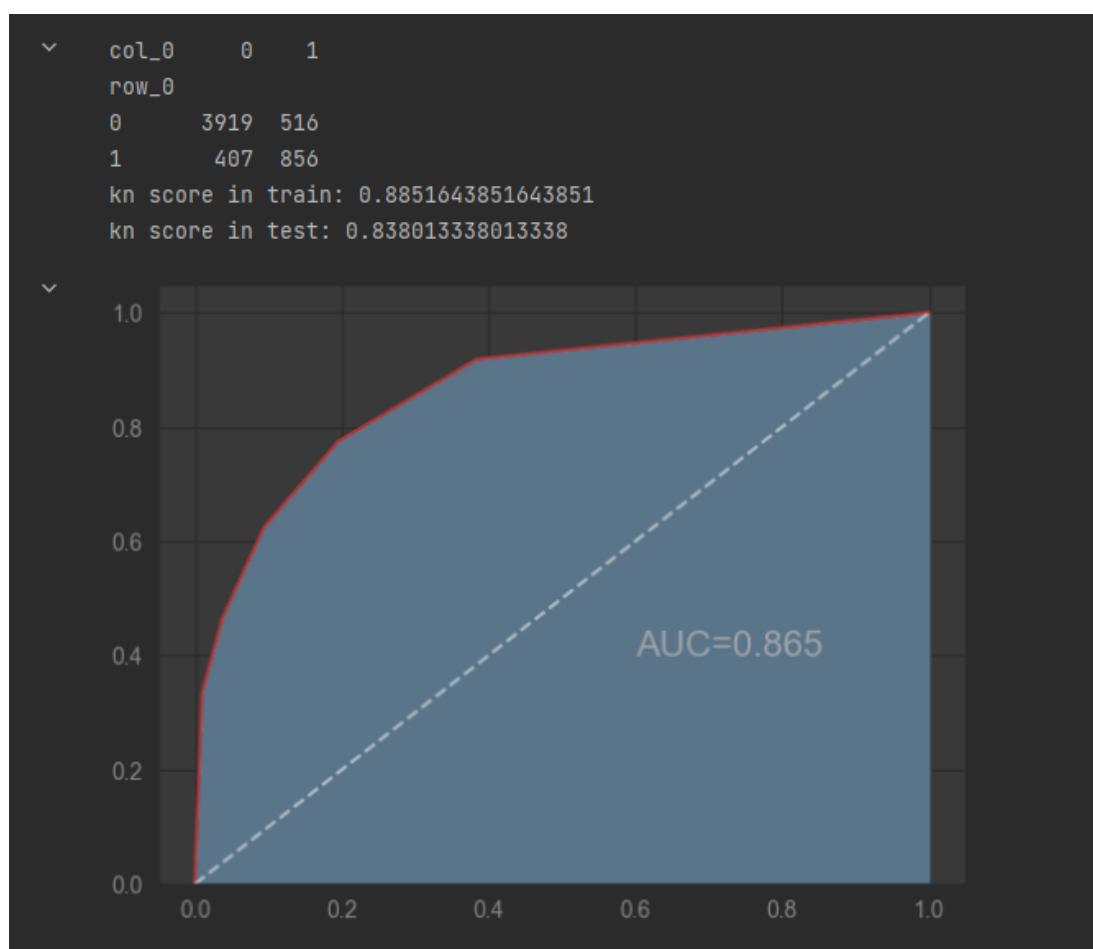
The code is as follows:

```
# 导入K近邻模型的类
from sklearn.neighbors import KNeighborsClassifier

# 构建K近邻模型
kn = KNeighborsClassifier()
kn.fit(trainx,trainy)
kn_pred = kn.predict(testx)
print(pd.crosstab(kn_pred, testy))
print("kn score in train:",kn.score(trainx,trainy))
print("kn score in test:",kn.score(testx,testy))

from sklearn import metrics
# 计算ROC曲线的x轴 和 y轴数据
fpr, tpr, _ = metrics.roc_curve(testy, kn.predict_proba(testx)[: ,1])
# 绘制ROC曲线
plt.plot(fpr, tpr, linestyle = 'solid', color = 'red')
# 添加阴影
plt.stackplot(fpr, tpr, color='steelblue')
# 绘制参考线
plt.plot([0,1],[0,1],linestyle='dashed', color='black')
# 添加文本
plt.text(0.6, 0.4, 'AUC=%.3f' % metrics.auc(fpr,tpr),
fontdict=dict(size =16))
plt.show()
```

4.2.2 Model Evaluation



The first part is the Confusion matrix. The rows in the matrix are the predicted values of the model, the columns in the matrix are the actual values of the test set, and the main diagonal is the number of correct predictions of the model.

After calculation, the conclusion in the second part is that the accuracy of the model in the training set is 88.5%, but the error rate on the test set exceeds 16% (1-0.838), indicating that there may be a risk of overfitting in the KNN model under default parameters.

The accuracy rate of the model is calculated based on the Confusion matrix, but this method has certain drawbacks, that is, if the data itself has a certain imbalance (the proportion of positive and negative samples varies greatly), the accuracy rate will be very high, but it does not necessarily mean that the model is ideal.

Therefore, the Receiver Operating Characteristic(ROC) can be drawn and the AUC value of the area under the curve can be calculated

The ROC of the model is drawn in the above figure. It is calculated that the AUC under the curve is 0.865. When using AUC to evaluate the quality of a model, it should be hoped that the larger the AUC, the better. Generally speaking, when the AUC value exceeds 0.8, it can be considered that the model is relatively reasonable. Therefore, the performance of the K-nearest neighbor model based on default parameters on the household income dataset is relatively ideal.

4.3 Gradient-Boosting-Decision-Tree

4.3.1 Model Establishment and Prediction

For the GBDT model, we can call the GradientBoostingClassifier class in the sklearn.ensemble, and also try to use the default parameters of the model.

First, that is, let the Learning rate (iteration step size) of the model be 0.1 (learning_rate=0.1), use the logarithmic Loss function (loss='deviance '), and generate 100 basic decision trees (n_estimators=100)

And the maximum depth of each basic decision tree is 3 (max_depth=3), the minimum sample size of the middle node (non leaf node) is 2 (min_samples_split=2), and the minimum sample size of the leaf node is 1 (min_samples_leaf=1).

The training of each tree will not be based on the results of the previous tree (warmStart=False).

```
plt.show()

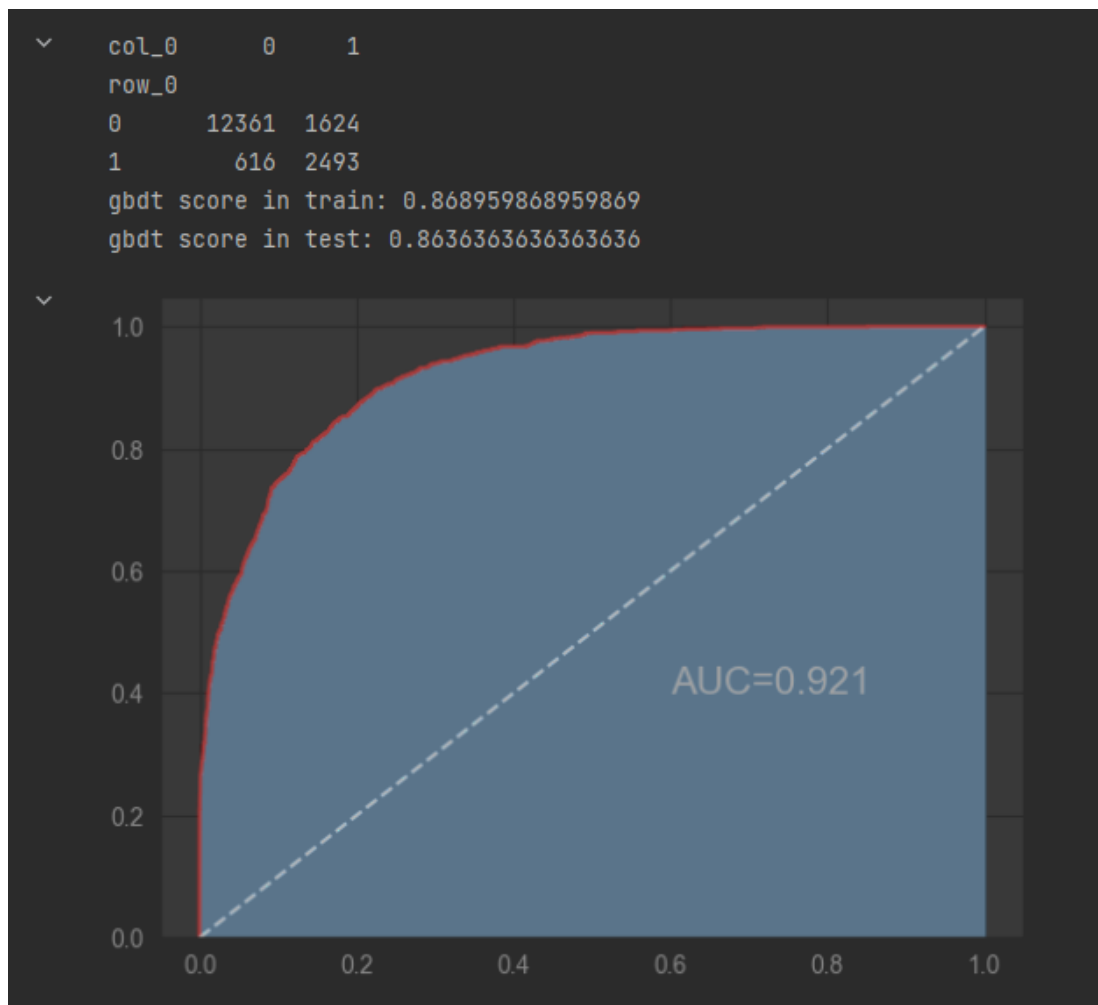
###
# 导入梯度提升决策树模型的类
from sklearn.ensemble import GradientBoostingClassifier
#构建GBDT模型
gbdt = GradientBoostingClassifier()
gbdt.fit(trainx, trainy)
gbdt_pred=gbdt.predict(trainx)
print(pd.crosstab(gbdt_pred, trainy))
print("gbdt score in train:",gbdt.score(trainx,trainy))
print("gbdt score in test:",gbdt.score(testx,testy))
from sklearn import metrics
# 计算ROC曲线的x轴 和 y轴数据
```

```

fpr, tpr, _ = metrics.roc_curve(testy, gbdtpredict_proba(testx)[: ,1])
# 绘制ROC曲线
plt.plot(fpr, tpr, linestyle = 'solid', color ='red')
# 添加阴影
plt.stackplot(fpr, tpr, color='steelblue')
# 绘制参考线
plt.plot([0,1],[0,1],linestyle='dashed', color='black')
# 添加文本
plt.text(0.6, 0.4, 'AUC=%.3f' % metrics.auc(fpr,tpr),
fontdict=dict(size =16))
plt.show()

```

4.3.2 Model Evaluation

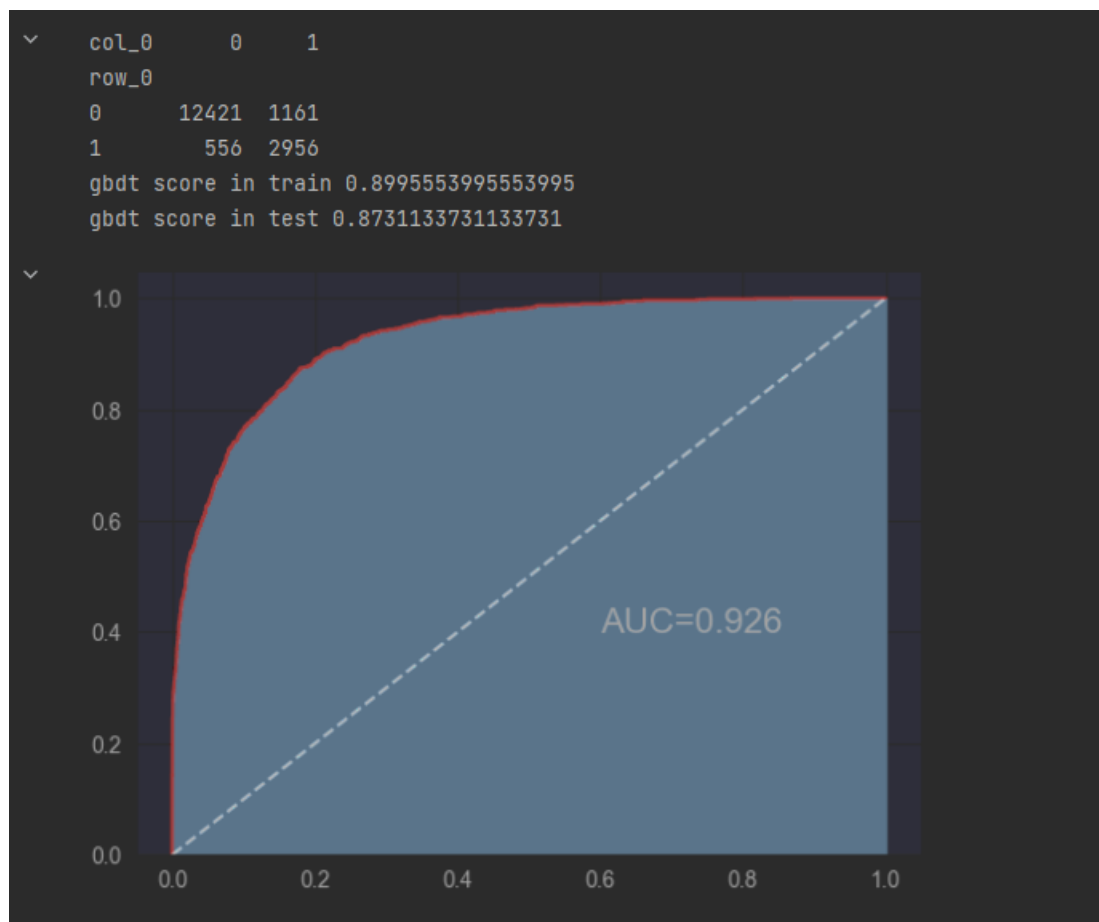


The performance of the integrated algorithm GBDT on the test set is significantly better than that of the K-nearest neighbor algorithm, which is the advantage of voting based on multiple decision trees.

The model performs very well on both the training and testing sets, with an accuracy rate of over 85%, and the AUC value is also the highest among the first three models, reaching 0.921.

4.3.3 Compare with One-Hot-Encoding GBDT

The following image shows the prediction results of the training and testing sets using a GBDT classifier using one-hot-encoding to encode discrete variables



From the above figure, we find that one-hot-encoding GBDT is more brilliant than the directly numerical-based values encoding.

No matter the accuracy in training sets and testing sets or the AUC values are all higher than the GBDT based on the directly numerical-based values encoding.

4.4 K-Nearest-Neighbour with GridSearchCV

4.4.1 Model Establishment and Prediction

The meanings of several parameters in the GridSearchCV function:

Estimator accepts a specified model, where is the class of the K-nearest neighbor model;

Param_Grid is used to specify the parameter list object that the model needs to search for, where n in the K-nearest neighbor model_11 possible values for the neighbors parameter;

CV refers to grid search that requires 10 times of cross validation;

Scoring specifies the measurement value for model evaluation, where the accuracy of model prediction is selected.

```
# K 近邻模型网格搜索法
# 导入网格搜索函数
from sklearn.model_selection import GridSearchCV
# 选择不同的参数
k_options = list(range(1,12))
parameters = {'n_neighbors':k_options}
# 搜索不同的K值
grid_kn = GridSearchCV(estimator= KNeighborsClassifier(),
param_grid=parameters, cv=10, scoring='accuracy')
grid_kn.fit(trainx, trainy)
grid_kn_pred=grid_kn.predict(trainx)
# 结果输出
print(grid_kn.scoring, grid_kn.best_params_, grid_kn.best_score_)
print(pd.crosstab(grid_kn_pred, trainy))
print("grid_kn score in train:",grid_kn.score(trainx,trainy))
print("grid_kn score in test:",grid_kn.score(testx,testy))
from sklearn import metrics
# 计算ROC曲线的x轴 和 y轴数据
fpr, tpr, _ = metrics.roc_curve(testy, grid_kn.predict_proba(testx)
[:,1])
# 绘制ROC曲线
```

```
plt.plot(fpr, tpr, linestyle = 'solid', color = 'red')
# 添加阴影
plt.stackplot(fpr, tpr, color='steelblue')
# 绘制参考线
plt.plot([0,1],[0,1],linestyle='dashed', color='black')
# 添加文本
plt.text(0.6, 0.4, 'AUC=%.3f' % metrics.auc(fpr,tpr),
fontdict=dict(size =16))
plt.show()
```

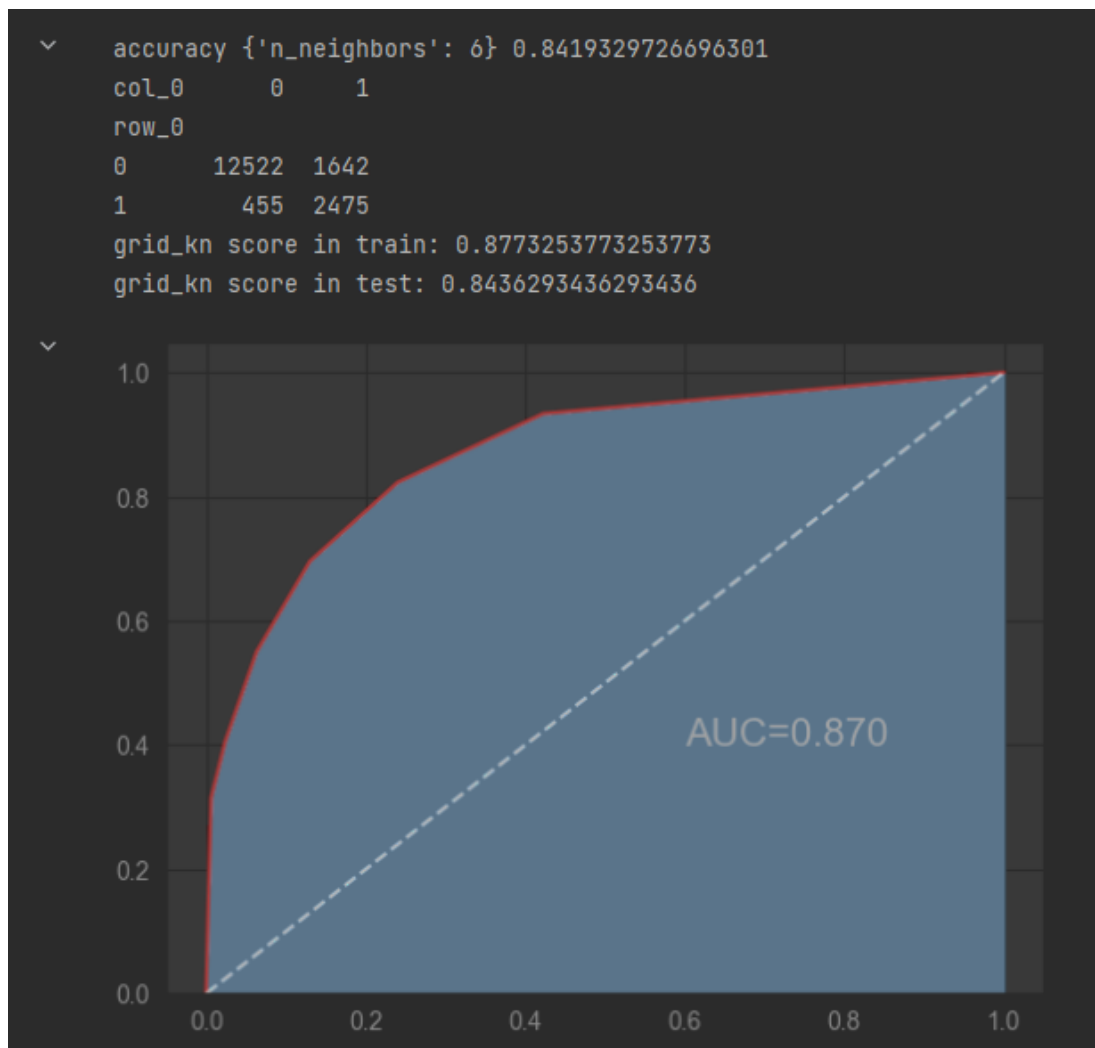
4.4.2 Model Evaluation

Through the calculation of grid search, three parts of results were obtained.

The first part includes the average accuracy under 11 K values (due to 10 cross validation).

The second part selected the optimal K value, with a K value of 6.

The third part is the optimal average accuracy of the model when the K value is 6, and the accuracy is 84.19%.



Next, we use the best parameters obtained from network search to predict the training set and dataset.

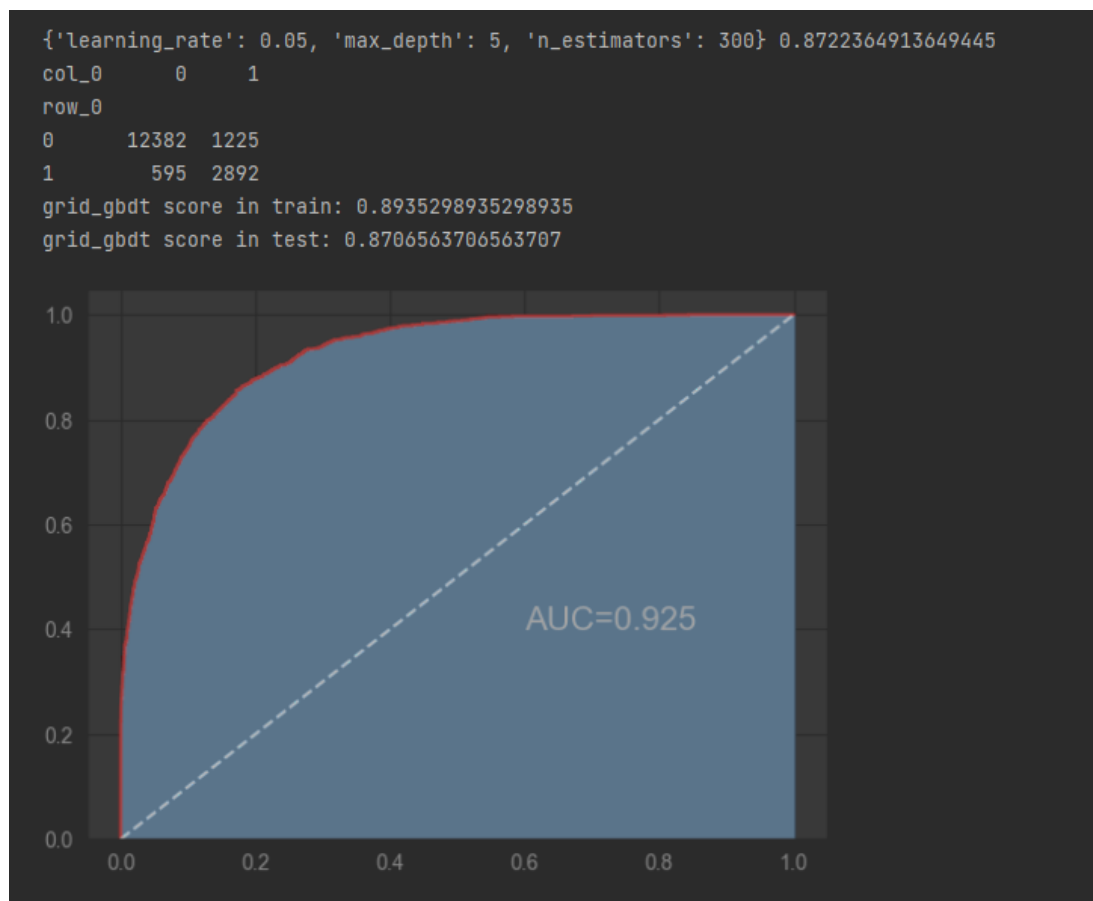
Compared to the K-nearest neighbor model with default parameters, the accuracy of the model after grid search on the training dataset decreased, but it improved on the test dataset, which is also what we expected. This indicates that the optimized model has better prediction performance, and the narrowing of the difference between the two can also reduce the possibility of overfitting of the model.

Then look at the area under the Receiver operating characteristic. The AUC corresponding to the K nearest neighbor model after grid search is 0.87, which is a little higher than the original KNN model. So, from the perspective of model stability, the K-nearest neighbor model after grid search is better than the original K-nearest neighbor model.

4.5 Gradient-Boosting-Decision-Tree with GridSearchCV

4.5.1 Model Establishment and Prediction

4.5.2 Model Evaluation



The performance of the GBDT model based on grid search is the best among the models in terms of accuracy, with an accuracy of nearly 89% on the training set and over 87% on the test set;

From the drawn Receiver operating characteristic, the AUC value is also the highest, exceeding 0.925.

Both the K-nearest neighbor model and the gradient lifting tree GBDT model can find their optimal model parameters through grid search, and the combination of these optimal parameters generally makes the model more excellent and robust.

So, vertically comparing the default parameter model with the best parameter model after grid search may be a better choice (although the latter may take more runtime); When comparing the single model and the integrated model horizontally, the integrated model will generally perform better than the single model.

5 Write Testlabel

```
path="..\testlabel.txt"
np.savetxt(path, test_label, fmt='%d')
```

6 Conclusion

6.1 Model Ranking

Model	Accuracy	AUC	Rank
Logistic Regression with One-Hot-Encoding	0.850	0.903	4
K-Nearest-Neighbour	0.838	0.865	6
Gradient-Boosting-Decision-Tree	0.869	0.921	3
K-Nearest-Neighbour with GridSearchCV	0.843	0.870	5
Gradient-Boosting-Decision-Tree with GridSearchCV	0.871	0.925	2
Naive_Bayes_GaussianNB	0.801	0.860	8
Naive_Bayes_BernoulliNB	0.713	0.768	10
Naive_Bayes_MultinomialNB	0.777	0.407	9
Gradient-Boosting-Decision-Tree with One-Hot-Encoding	0.873	0.926	1
Decision-Tree	0.805	0.749	7

From the above table, among the 10 models we build, the models which involve Gradient-Boosting-Decision-Tree perform best with accuracy beyond 87%. And the K-Nearest-Neighbour and the Logistic Regression with One-Hot-Encoding models also perform well.

6.2 Inspiration and Thinking

- Different data preprocessing methods can result in different outcomes, such as the one-hot-encoding and the numerical-based values encoding. And the one-hot-encoding seems to be better.
- I am more familiar with the important processes of data mining (previewing datasets, clarifying the purpose of analysis - importing data and data preprocessing - exploring data features - cleaning data, building models - model prediction, model evaluation)
- We can improve the model parameter optimization model using grid search method.
- I gain a deeper insight into classification, machine learning and all kinds of models.