《数据科学导论》课程期末作业

（20213385 张昕然 计算机 2106 班）

一、 随机种子的指定和数据预处理

1.1 代码实现及解释

（1）输入学号，生成随机数种子

```
random.seed(20213385) #设置学号
```

（2）限定 CSV 文件的表头

```
header = ["id", "age", "workclass", "fnlwgt", "education", "education-num", "marital-status", "occupation", "relationship","race", "sex", "capital-gain", "capital-loss", "h
```

（3）处理数据

步骤：

1） 初始化正负样例列表
2） 将行、元素进行分割
3） 将大于 50k 的样本和小于 50k 的样本分别放入正样本列表和负样本列表
4） 将正负样本随机打乱后存入 CSV 文件中
5） 打印各个样本的条数

```
def process_data1(inpath, outpath, nline):
    pos_data = list()
    neg_data = list()
    with open(inpath) as fin:
        for line in fin:
            line = line.strip().strip(".")
            tokens = line.split(", ")
            if tokens[-1].strip() == "<=50K":
                neg_data.append(line)
            elif tokens[-1].strip() == ">50K":
                pos_data.append(line)
    with open(outpath + ".csv", "w") as fout:
        csv_writer = csv.writer(fout)
        csv_writer.writerow(header)
        random.shuffle(pos_data)
        random.shuffle(neg_data)
        print(len(pos_data))
        print(len(neg_data))
        data = pos_data[:nline] + neg_data[:nline]
        for step, line in enumerate(data):
            tokens = [str(step+1)] + line.split(", ")
            csv_writer.writerow(tokens)
```

1.2 运行结果

控制台中打印各个样本的条数且在原文件夹中生成了对应的 CSV 文件。

```
7841
24720
3846
12435
```

二、 在 Income 数据集上建立模型

2.1 代码实现及解释

```python
# 步骤 1：导入必要的库
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import accuracy_score, recall_score, f1_score, roc_auc_score
import pandas as pd
import statsmodels.api as sm  # 导入statsmodels
from sklearn.model_selection import StratifiedKFold
from sklearn.linear_model import LogisticRegression
```

这里导入了用于数据处理和建模的一些常用库，包括 Scikit-Learn 和 Statsmodels。

```python
# 步骤 2：定义函数进行数据处理
2 用法
def preprocess_data(file_path):
    header = ["age", "workclass", "fnlwgt", "education", "education-num", "marital-status", "occupation",
              "relationship",
              "race", "sex", "capital-gain", "capital-loss", "hours-per-week", "native-country", "label"]

    data = pd.read_csv(file_path, names=header, delimiter=',', skipinitialspace=True)
    data.replace("?", pd.NA, inplace=True)

    missing_values_count = data[data.isnull().any(axis=1)].shape[0]

    if missing_values_count / data.size <= 0.1:
        print("不超过10%，因此删除带有缺失值的行")
        data.dropna(inplace=True)
    else:
        data.fillna(data.mean(), inplace=True)
        print("超过10%，因此用均值填充")

    X = data.drop("label", axis=1)
    Y = data["label"]

    Y = (Y == ">50K")

    categorical_cols = X.select_dtypes(include='object').columns
    X = pd.get_dummies(X, columns=categorical_cols)

    X = X.astype(float)
    return X, Y
```

该函数用于读取数据文件，处理缺失值（删除或填充均值），进行独热编码，最终返回特征矩阵 X 和目标变量 Y。

```python
# 步骤 3：定义函数进行模型训练
1 个用法
def train_model(X, Y):
    X = sm.add_constant(X)
    model = sm.Logit(Y, X)
    #result_model = model.fit()
    #result_model = model.fit_regularized(method='l1')  # 或 method='l2'
    result_model = model.fit()  # 或 method='l2'
    return result_model
```

该函数将特征矩阵 X 和目标变量 Y 传入 Logistic Regression 模型中进行拟合，并返回训练好的模型。

```python
# 步骤 4：定义函数提取模型系数
1 个用法
def extract_coefficients(features, coefficients):
    for feature, coefficient in zip(features, coefficients):
        print(f"{feature}: {coefficient}")
```

该函数接收特征和对应的系数，然后将它们打印出来。

```python
# 步骤 5：定义函数获取模型摘要
1 个用法
def get_model_summary(result_model):
    model_summary = result_model.summary()
    print(model_summary)
```

该函数接收训练好的 Logistic Regression 模型，然后输出模型的摘要信息，包括系数、标准误差、z 值等。

```python
# 步骤 6：定义函数进行交叉验证
1 个用法
def perform_cross_validation(X, Y, cv=5):
    # 使用StratifiedKFold来保持类别分布
    kfold = StratifiedKFold(n_splits=cv, shuffle=True, random_state=42)

    # 初始化逻辑回归模型
    logistic_model = LogisticRegression(max_iter=2000)

    # 交叉验证
    scores1 = cross_val_score(logistic_model, X, Y, cv=kfold, scoring='accuracy')
    scores2 = cross_val_score(logistic_model, X, Y, cv=kfold, scoring='recall')

    # 输出交叉验证准确率
    print(f"Cross-Validation Accuracy: {scores1.mean()}")
    print(f"Cross-Validation recall: {scores2.mean()}")
```

该函数使用 StratifiedKFold 进行交叉验证，初始化 Logistic Regression 模型，然后输出交叉验证的准确率和召回率的平均值。

```
# 步骤 7：定义函数进行模型预测和性能指标计算
1 个用法
def evaluate_model(result_model, test_X, test_Y, threshold=0.5):
    # 对测试数据进行与训练数据相同的处理
    test_X.replace("?", pd.NA, inplace=True)

    missing_values_count = test_X[test_X.isnull().any(axis=1)].shape[0]

    if missing_values_count / test_X.size <= 0.1:
        print("Not exceeding 10%, so delete rows with missing values")
        test_X.dropna(inplace=True)
    else:
        test_X.fillna(test_X.mean(), inplace=True)
        print("Exceeding 10%, so fill with mean")

    # 独热编码
    categorical_cols_test = test_X.select_dtypes(include='object').columns
    test_X = pd.get_dummies(test_X, columns=categorical_cols_test)

    # 确保测试数据与训练数据具有相同的列，缺失的列补0
    missing_cols = list(set(train_X.columns) - set(test_X.columns))
    test_X = pd.concat([test_X, pd.DataFrame(0, index=test_X.index, columns=missing_cols)], axis=1)

    # 确保列的顺序一致
    test_X = test_X[train_X.columns]

    # 转换数据类型为数值型
    test_X = test_X.astype(float)

    # 使用已训练的模型进行预测
    test_X = sm.add_constant(test_X)
    predictions = result_model.predict(test_X)
```

```
# 计算性能指标
predicted_labels = predictions > threshold

accuracy = accuracy_score(test_Y, predicted_labels)
recall = recall_score(test_Y, predicted_labels, zero_division=1)
f1 = f1_score(test_Y, predicted_labels, zero_division=1)

# ROC AUC Score 处理异常情况
try:
    roc_auc = roc_auc_score(test_Y, predictions)
    print(f"ROC AUC Score: {roc_auc}")
except ValueError:
    print("ROC AUC Score is not defined in this case.")

# 展示性能指标
print(f"Accuracy: {accuracy}")
print(f"Recall: {recall}")
print(f"F1 Score: {f1}")
```

该函数接收训练好的模型、测试集特征和目标变量，进行与训练数据相同的预处理，然

后使用模型进行预测，计算并打印性能指标，如准确率、召回率、F1 分数和 ROC AUC 分数。

```
# 步骤 8：应用上述函数进行训练和测试
train_file_path = 'Income.data'
test_file_path = 'answer.csv'
#test_file_path = 'Income.data'

# 训练数据处理
train_X, train_Y = preprocess_data(train_file_path)

# 模型训练
result_model = train_model(train_X, train_Y)

# 提取系数
extract_coefficients(train_X.columns, result_model.params)

# 获取模型摘要
get_model_summary(result_model)

# 测试数据处理
test_X, test_Y = preprocess_data(test_file_path)


# 模型预测和性能指标计算
evaluate_model(result_model, test_X, test_Y)


#进行交叉验证
perform_cross_validation(train_X, train_Y)
```

最后，应用上述函数进行整个流程：训练模型、提取系数、获取模型摘要、测试数据预处理、模型预测和性能指标计算，以及进行交叉验证。

三、 在 salary.test 数据集验证效果

3.1 模型效果

经过运行模型和评估代码，得到了模型的准确率和召回率，分别为：Accuracy0.7586512866015972、Recall0.6061549100968188。（阈值为 0.5）

```
Accuracy: 0.7586512866015972
Recall: 0.6061549100968188
```

（1）准确率（Accuracy):

准确率是模型正确预测的样本数量与总样本数量之比，反映了整体预测的准确性。

在这个场景中，模型的准确率为 0.7587，即约为 75.87%。

高准确率表明模型在整体上具有较好的分类性能，正确预测了大部分样本。

（2）召回率（Recall):

召回率是所有真实正例中被模型正确识别为正例的比例，衡量了模型对正例的覆盖程度。

在这个场景中，模型的召回率为 0.6062，即约为 60.62%。

相对较高的召回率意味着模型能够较好地捕捉真实正例，对于识别目标类别的能力较强。

3.2 交叉验证

经过运行模型和交叉验证代码，得到了模型的交叉验证准确率和交叉验证召回率，分别为：Cross-Validation Accuracy0.790763224897898、Cross-Validation recall: 0.26558423988978497。（阈值为 0.5）

```
Cross-Validation Accuracy: 0.790763224897898
Cross-Validation recall: 0.26558423988978497
```

（1）交叉验证准确率（Cross-Validation Accuracy):

交叉验证准确率为 0.7908，即约为 79.08%。

相较于单一模型的准确率，交叉验证准确率稍高，说明模型在不同的训练-验证集划分上能够保持一定的稳定性。

（2）交叉验证召回率（Cross-Validation Recall):

交叉验证召回率为 0.2656，即约为 26.56%。

与之前的单一模型召回率相比，交叉验证的召回率明显下降。这可能是由于交叉验证使用了多个不同的训练集和验证集，导致模型在捕捉正例时的一致性稍降。

（3）综合分析：

交叉验证的准确率相对较高，说明模型在不同数据子集上仍然能够较为准确地进行分类预测。

交叉验证的召回率较低，这可能表明模型在不同数据子集上对于捕捉真实正例的能力较为不稳定，存在一定的波动性。

3.3 过拟合性

分别在 income.data（训练集）上和 answer.csv（测试集）上对模型性能进行评估，得到各自的准确率和召回率，通过比较两组数据，可以分析过拟合的情况。

通过实验得到：

Accuracy: 0.7586512866015972 Recall: 0.6061549100968188 （以上是测试集结果）

Accuracy: 0.8498110204893574 Recall: 0.6109483217900906 （以上是训练集结果）

测试集表现（Accuracy: 0.7587, Recall: 0.6062）：

1.测试集的准确率和召回率相对较低，可能暗示模型在未见过的数据上的泛化能力较为有限。

2.准确率和召回率之间存在一定的差距，这可能表明模型在测试集上并没有很好地捕捉到所有的真实正例。

训练集表现（Accuracy: 0.8498, Recall: 0.6109）：

1.训练集上的性能相对较好，准确率和召回率都较高。

2.与测试集相比，训练集上的性能表现更为优越，说明模型可能在训练数据上有过拟合现象。

过拟合性分析：

1.过拟合通常表现为模型在训练集上表现良好，但在未见过的测试集上性能下降。

2.在这里，测试集上的准确率和召回率相对较低，与训练集相比存在差异。

3.准确率和召回率之间的差距可能暗示了模型对于训练数据中的噪声或样本特定性的过拟合。

4.由于训练集和测试集的评估结果相差并不大，故认为有过拟合现象，但是在可以接受的范围内，并不严重。

四、 用 salary.data 数据集训练模型

4.1 操作方法

将原本输入到数据处理模块并最终输入模型训练的数据集改为 salary.data(deal.csv)。即可改变训练数据集。

```
# 步骤 7：应用上述函数进行训练和测试
train_file_path = 'deal.csv'
test_file_path = 'answer.csv'
```

4.2 模型效果

经过运行模型和评估代码，得到了模型的准确率和召回率，分别为：Accuracy: 0.8181011535048802、Recall: 0.8454356846473029。（阈值为 0.5）

```
Accuracy: 0.8181011535048802
Recall: 0.8454356846473029
```

可见，模型效果在 salary.data 数据集上有了明显提升。

下图为原来数据集时的测试数据：

```
Cross-Validation Accuracy: 0.790763224897898
Cross-Validation recall: 0.26558423988978497
```

4.3 交叉验证

经过运行模型和交叉验证代码，得到了模型的交叉验证准确率和交叉验证召回率，分别为：Cross-Validation Accuracy: 0.6291960841518364 、Cross-Validation recall: 0.5784763426542388。（阈值为 0.5）

```
Cross-Validation Accuracy: 0.6291960841518364
Cross-Validation recall: 0.5784763426542388
```

（1）交叉验证准确率（Cross-Validation Accuracy):

交叉验证准确率为 0.6291，即约为 62.91%。

相比准确率下降，可能是由于可用来训练和

（2）交叉验证召回率（Cross-Validation Recall):

交叉验证召回率为 0.2656，即约为 26.56%。

与之前的单一模型召回率相比，交叉验证的召回率明显下降。这可能是由于交叉验证使用了多个不同的训练集和验证集，导致模型在捕捉正例时的一致性稍降。

（3）综合分析:

交叉验证的准确率相对较高，说明模型在不同数据子集上仍然能够较为准确地进行分类预测。

交叉验证的召回率较低，这可能表明模型在不同数据子集上对于捕捉真实正例的能力较为不稳定，存在一定的波动性。

4.4 原因分析

在经过正负例平衡的样本训练的模型与原本完整数据集的模型之间存在性能差异可能涉及到数据不平衡对模型训练的影响。经过正负例平衡处理的训练集在样本数量上对正负类别进行了均衡，这有助于模型更好地学习到不同类别的特征和模式，从而在测试集上表现更好。

经过正负例平衡的模型在原本完整数据集上表现更好的准确率和召回率，这可能是因为原始数据中存在着类别不平衡，使得模型更偏向于预测样本数量更多的类别。通过平衡样本，模型更能够充分利用正负类别的信息，提高了对较少样本的类别的学习能力，从而提升了模型在原始数据集上的性能。

交叉验证的结果却呈现出不同的情况。在交叉验证中，正负例平衡的模型准确率较低，但召回率却较高。这可能是因为在交叉验证的过程中，模型在每个折叠中都要面对不同的训练集和验证集，导致模型对正例的识别更为一致，而准确率则受到不平衡类别分布的影响。

五、 刻画模型

5.1 coefficient 系数

通过 result_model.params 函数，可以得到所有经过处理后留下的变量 coefficient 系数，如下：

age: -7.216090019012195
fnlwgt: 0.025497948797778665
education-num: 7.515463211985386e-07
capital-gain: 0.5215072531854577
capital-loss: 0.00032253281858231374
hours-per-week: 0.0006420211962023044
workclass_Federal-gov: 0.029492647949003276
workclass_Local-gov: 3.2223521240172075
workclass_Private: 2.5238415301340558
workclass_Self-emp-inc: 2.716873740088045
workclass_Self-emp-not-inc: 2.89303064714745
workclass_State-gov: 2.225122977833857
workclass_Without-pay: 2.4016039657820234
education_10th: -23.19891594689341
education_11th: 1.357436531793972
education_12th: 0.9305475589936798
education_1st-4th: 0.7587237337784226
education_5th-6th: 3.003650452226971
education_7th-8th: 2.5263789504805887
education_9th: 1.8364584729602182

education_Assoc-acdm: 1.641734625182582
education_Assoc-voc: -0.5021542548981452
education_Bachelors: 0.017973997034724232
education_Doctorate: -0.3943640611128351
education_HS-grad: -0.9224921073038211
education_Masters: 0.566422917019224
education_Preschool: -0.5557806155233878
education_Prof-school: -17.368807359992356
education_Some-college: -0.49255772986401647
marital-status_Divorced: 0.38073869352196055
marital-status_Married-AF-spouse: -1.6730052643351887
marital-status_Married-civ-spouse: 1.0947442423705847
marital-status_Married-spouse-absent: 0.43226299655533784
marital-status_Never-married: -1.6608012238131564
marital-status_Separated: -2.1590834608231297
marital-status_Widowed: -1.7624009163298244
occupation_Adm-clerical: -1.4878083063762266
occupation_Armed-Forces: -0.14198911714141235
occupation_Craft-repair: -1.3066307424238657
occupation_Exec-managerial: -0.07830107512610586
occupation_Farming-fishing: 0.6634269042812061
occupation_Handlers-cleaners: -1.1228626540834474
occupation_Machine-op-inspct: -0.836988717111555
occupation_Other-service: -0.40529270600246237
occupation_Priv-house-serv: -0.9665048584710579
occupation_Prof-specialty: -4.294694864428298
occupation_Protective-serv: 0.37454942453106166
occupation_Sales: 0.45583186990518637
occupation_Tech-support: 0.15233849167915825
occupation_Transport-moving: 0.5228329661672392
relationship_Husband: -0.23180715219204504
relationship_Not-in-family: -1.371175706512984
relationship_Other-relative: -0.9189951519590785
relationship_Own-child: -1.7671780450033063
relationship_Unmarried: -2.1033578707685616
relationship_Wife: -1.0353316314449528
race_Amer-Indian-Eskimo: -0.020052618070869256
race_Asian-Pac-Islander: -1.8386163639257287
race_Black: -1.0105977212900281
race_Other: -1.4026936914006765
race_White: -1.713080993746719
sex_Female: -1.2511020376502155
sex_Male: -4.040455921615796
native-country_Cambodia: -3.1756348802835954

native-country_Canada: 2.164204161504375
native-country_China: 1.3528593491207055
native-country_Columbia: 0.24848512720366614
native-country_Cuba: -1.1107924569238579
native-country_Dominican-Republic: 1.3904181309537251
native-country_Ecuador: -0.75048227109458
native-country_El-Salvador: 0.7638318858066362
native-country_England: 0.41916640861277604
native-country_France: 1.3293611302004993
native-country_Germany: 1.6038500765863228
native-country_Greece: 1.478230604276783
native-country_Guatemala: 0.03847969567958756
native-country_Haiti: 0.7685178500239235
native-country_Holand-Netherlands: 0.9948645392878211
native-country_Honduras: -19.999661969130234
native-country_Hong: -0.14213352489982883
native-country_Hungary: 0.8087591400622879
native-country_India: 0.9106440284411519
native-country_Iran: 0.49983138113581066
native-country_Ireland: 1.0412150464289798
native-country_Italy: 1.5484423558775273
native-country_Jamaica: 1.834734508562753
native-country_Japan: 1.0390001461089764
native-country_Laos: 1.2229280694073636
native-country_Mexico: 0.28162488856347484
native-country_Nicaragua: 0.5150370341464926
native-country_Outlying-US(Guam-USVI-etc): 0.284165742313058
native-country_Peru: -16.85757566498949
native-country_Philippines: 0.1796054352305358
native-country_Poland: 1.2860268047173065
native-country_Portugal: 1.018197436580931
native-country_Puerto-Rico: 1.0425763383331845
native-country_Scotland: 0.7246788553062622
native-country_South: 0.7571758824908504
native-country_Taiwan: -0.2819520128730333
native-country_Thailand: 0.7803855179477127
native-country_Trinadad&Tobago: 0.3333692111873471
native-country_United-States: 0.5839290956389572
native-country_Vietnam: 1.2093201144910657
native-country_Yugoslavia: -0.2307344000270013

在系数（coefficients）的分析中，可以关注一些具有显著影响的特征，这些系数表示了模型对于不同特征的权重。以下是一些具有较大系数的特征，以及它们对模型的影响。

1. education_9th (1.8365):

正的系数表明拥有 9 年级教育水平的个体更有可能属于高收入群体。这突显了教育对收入的潜在影响，即使在较低的教育水平下也是如此。

2. marital-status_Separated (-2.1591):

负的系数表示已分居的个体更不可能属于高收入群体。婚姻状况似乎影响收入水平，已分居的个体更不可能有更高的收入。

3. occupation_Farming-fishing (0.6634):

正的系数意味着从事农业或渔业职业的个体更有可能属于高收入群体。这反映了职业对收入的影响，某些领域与较高的收入相关。

4. relationship_Wife (-1.0353):

负的系数表明被标识为“Wife”的个体更不可能拥有超过 50K 的收入。这强调了婚姻状况在预测收入中的作用，妻子更不可能有更高的收入。

5. race_Asian-Pac-Islander (-1.8386):

负的系数表示属于亚太岛民族的个体更不可能拥有超过 50K 的收入。这指出了收入水平上的种族差异，该群体更不可能有更高的收入。

6. native-country_Canada (2.1642):

正的系数表明来自加拿大的个体更有可能属于高收入群体。这意味着原籍国对收入水平可能存在影响。

7. native-country_United-States (0.5839):

正的系数表示来自美国的个体更有可能拥有超过 50K 的收入。这强调了国籍对收入的影响，美国居民更有可能有更高的收入。

8. occupation_Protective-serv (0.3745):

正的系数意味着从事保护服务职业的个体更有可能属于高收入群体。这表明该类别内的某些职业可能与更高的收入相关。

9. capital-loss (0.0003):

资本损失的正系数表明，令人惊讶的是，较高的资本损失与更高的可能性有关。可能需要进一步调查以了解这种意外的关系。

10. age (-7.2161):

年龄的负系数表示年龄较大的个体更不可能属于高收入群体。这挑战了随着年龄增长收入增加的普遍假设，表明年龄与收入之间存在微妙的关系。

5.2 summary 摘要

通过 result_model.summary()函数，可以得到所有经过处理后留下的变量 summary 摘要，如下：

Logit Regression Results

Dep. Variable:	label	No. Observations:	30162
Model:	Logit	Df Residuals:	30066
Method:	MLE	Df Model:	95
Date:	Mon, 08 Jan 2024	Pseudo R-squ.:	0.4244
Time:	19:48:15	Log-Likelihood:	-9742.9
converged:	False	LL-Null:	-16925.
Covariance Type:	nonrobust	LLR p-value:	0.000

	coef	std err	z	P>|z|	[0.025	0.975]
const	-7.2161	nan	nan	nan	nan	nan
age	0.0255	0.002	14.890	0.000	0.022	0.029
fnlwgt	7.515e-07	1.76e-07	4.264	0.000	4.06e-07	1.1e-06
education-num	0.5215	nan	nan	nan	nan	nan
capital-gain	0.0003	1.07e-05	30.022	0.000	0.000	0.000
capital-loss	0.0006	3.85e-05	16.696	0.000	0.001	0.001
hours-per-week	0.0295	0.002	17.325	0.000	0.026	0.033
workclass_Federal-gov	3.2224	nan	nan	nan	nan	nan
workclass_Local-gov	2.5238	nan	nan	nan	nan	nan
workclass_Private	2.7169	nan	nan	nan	nan	nan
workclass_Self-emp-inc	2.8930	nan	nan	nan	nan	nan
workclass_Self-emp-not-inc	2.2251	nan	nan	nan	nan	nan
workclass_State-gov	2.4016	2.8e+05	8.59e-06	1.000	-5.48e+05	5.48e+05
workclass_Without-pay	-23.1989	2.58e+05	-9.01e-05	1.000	-5.05e+05	5.05e+05
education_10th	1.3574	nan	nan	nan	nan	nan
education_11th	0.9305	nan	nan	nan	nan	nan
education_12th	0.7587	nan	nan	nan	nan	nan
education_1st-4th	3.0037	nan	nan	nan	nan	nan
education_5th-6th	2.5264	nan	nan	nan	nan	nan
education_7th-8th	1.8365	nan	nan	nan	nan	nan
education_9th	1.6417	nan	nan	nan	nan	nan
education_Assoc-acdm	-0.5022	nan	nan	nan	nan	nan
education_Assoc-voc	0.0180	nan	nan	nan	nan	nan
education_Bachelors	-0.3944	nan	nan	nan	nan	nan
education_Doctorate	-0.9225	nan	nan	nan	nan	nan
education_HS-grad	0.5664	nan	nan	nan	nan	nan
education_Masters	-0.5558	nan	nan	nan	nan	nan
education_Preschool	-17.3688	nan	nan	nan	nan	nan
education_Prof-school	-0.4926	nan	nan	nan	nan	nan
education_Some-college	0.3807	nan	nan	nan	nan	nan
marital-status_Divorced	-1.6730	1.19e+06	-1.4e-06	1.000	-2.34e+06	2.34e+06
marital-status_Married-AF-spouse	1.0947	1.22e+06	8.95e-07	1.000	-2.4e+06	2.4e+06
marital-status_Married-civ-spouse	0.4323	1.23e+06	3.52e-07	1.000	-2.41e+06	2.41e+06
marital-status_Married-spouse-absent	-1.6608	1.18e+06	-1.4e-06	1.000	-2.32e+06	2.32e+06
marital-status_Never-married	-2.1591	1.24e+06	-1.75e-06	1.000	-2.42e+06	2.42e+06
marital-status_Separated	-1.7624	1.23e+06	-1.43e-06	1.000	-2.41e+06	2.41e+06
marital-status_Widowed	-1.4878	1.27e+06	-1.17e-06	1.000	-2.48e+06	2.48e+06
occupation_Adm-clerical	-0.1420	1.58e+06	-8.99e-08	1.000	-3.1e+06	3.1e+06
occupation_Armed-Forces	-1.3066	1.61e+06	-8.14e-07	1.000	-3.15e+06	3.15e+06
occupation_Craft-repair	-0.0783	1.59e+06	-4.94e-08	1.000	-3.11e+06	3.11e+06
occupation_Exec-managerial	0.6634	1.6e+06	4.14e-07	1.000	-3.14e+06	3.14e+06

occupation_Farming-fishing	-1.1229	1.58e+06	-7.12e-07	1.000	-3.09e+06	3.09e+06
occupation_Handlers-cleaners	-0.8370	1.6e+06	-5.24e-07	1.000	-3.13e+06	3.13e+06
occupation_Machine-op-inspct	-0.4053	1.62e+06	-2.51e-07	1.000	-3.17e+06	3.17e+06
occupation_Other-service	-0.9665	1.61e+06	-6e-07	1.000	-3.16e+06	3.16e+06
occupation_Priv-house-serv	-4.2947	1.61e+06	-2.66e-06	1.000	-3.16e+06	3.16e+06
occupation_Prof-specialty	0.3745	1.59e+06	2.35e-07	1.000	-3.12e+06	3.12e+06
occupation_Protective-serv	0.4558	1.58e+06	2.88e-07	1.000	-3.1e+06	3.1e+06
occupation_Sales	0.1523	1.59e+06	9.61e-08	1.000	-3.11e+06	3.11e+06
occupation_Tech-support	0.5228	1.6e+06	3.27e-07	1.000	-3.13e+06	3.13e+06
occupation_Transport-moving	-0.2318	1.59e+06	-1.46e-07	1.000	-3.12e+06	3.12e+06
relationship_Husband	-1.3712	1e+06	-1.37e-06	1.000	-1.97e+06	1.97e+06
relationship_Not-in-family	-0.9190	9.95e+05	-9.23e-07	1.000	-1.95e+06	1.95e+06
relationship_Other-relative	-1.7672	9.75e+05	-1.81e-06	1.000	-1.91e+06	1.91e+06
relationship_Own-child	-2.1034	9.94e+05	-2.12e-06	1.000	-1.95e+06	1.95e+06
relationship_Unmarried	-1.0353	9.42e+05	-1.1e-06	1.000	-1.85e+06	1.85e+06
relationship_Wife	-0.0201	1e+06	-2e-08	1.000	-1.96e+06	1.96e+06
race_Amer-Indian-Eskimo	-1.8386	nan	nan	nan	nan	nan
race_Asian-Pac-Islander	-1.0106	nan	nan	nan	nan	nan
race_Black	-1.4027	nan	nan	nan	nan	nan
race_Other	-1.7131	nan	nan	nan	nan	nan
race_White	-1.2511	nan	nan	nan	nan	nan
sex_Female	-4.0405	1.74e+06	-2.32e-06	1.000	-3.42e+06	3.42e+06
sex_Male	-3.1756	1.74e+06	-1.83e-06	1.000	-3.41e+06	3.41e+06
native-country_Cambodia	2.1642	7.61e+05	2.84e-06	1.000	-1.49e+06	1.49e+06
native-country_Canada	1.3529	7.61e+05	1.78e-06	1.000	-1.49e+06	1.49e+06
native-country_China	0.2485	7.61e+05	3.26e-07	1.000	-1.49e+06	1.49e+06
native-country_Columbia	-1.1108	7.61e+05	-1.46e-06	1.000	-1.49e+06	1.49e+06
native-country_Cuba	1.3904	7.61e+05	1.83e-06	1.000	-1.49e+06	1.49e+06
native-country_Dominican-Republic	-0.7505	7.61e+05	-9.86e-07	1.000	-1.49e+06	1.49e+06
native-country_Ecuador	0.7638	7.61e+05	1e-06	1.000	-1.49e+06	1.49e+06
native-country_El-Salvador	0.4192	7.61e+05	5.51e-07	1.000	-1.49e+06	1.49e+06
native-country_England	1.3294	7.61e+05	1.75e-06	1.000	-1.49e+06	1.49e+06
native-country_France	1.6039	7.61e+05	2.11e-06	1.000	-1.49e+06	1.49e+06
native-country_Germany	1.4782	7.61e+05	1.94e-06	1.000	-1.49e+06	1.49e+06
native-country_Greece	0.0385	7.61e+05	5.05e-08	1.000	-1.49e+06	1.49e+06
native-country_Guatemala	0.7685	7.61e+05	1.01e-06	1.000	-1.49e+06	1.49e+06
native-country_Haiti	0.9949	7.61e+05	1.31e-06	1.000	-1.49e+06	1.49e+06
native-country_Holand-Netherlands	-19.9997	8.12e+05	-2.46e-05	1.000	-1.59e+06	1.59e+06
native-country_Honduras	-0.1421	7.61e+05	-1.87e-07	1.000	-1.49e+06	1.49e+06
native-country_Hong	0.8088	7.61e+05	1.06e-06	1.000	-1.49e+06	1.49e+06
native-country_Hungary	0.9106	7.61e+05	1.2e-06	1.000	-1.49e+06	1.49e+06
native-country_India	0.4998	7.61e+05	6.56e-07	1.000	-1.49e+06	1.49e+06
native-country_Iran	1.0412	7.61e+05	1.37e-06	1.000	-1.49e+06	1.49e+06
native-country_Ireland	1.5484	7.61e+05	2.03e-06	1.000	-1.49e+06	1.49e+06

native-country_Italy	1.8347	7.61e+05	2.41e-06	1.000	-1.49e+06	1.49e+06
native-country_Jamaica	1.0390	7.61e+05	1.36e-06	1.000	-1.49e+06	1.49e+06
native-country_Japan	1.2229	7.61e+05	1.61e-06	1.000	-1.49e+06	1.49e+06
native-country_Laos	0.2816	7.61e+05	3.7e-07	1.000	-1.49e+06	1.49e+06
native-country_Mexico	0.5150	7.61e+05	6.76e-07	1.000	-1.49e+06	1.49e+06
native-country_Nicaragua	0.2842	7.61e+05	3.73e-07	1.000	-1.49e+06	1.49e+06
native-country_Outlying-US(Guam-USVI-etc)	-16.8576	7.61e+05	-2.21e-05	1.000	-1.49e+06	1.49e+06
native-country_Peru	0.1796	7.61e+05	2.36e-07	1.000	-1.49e+06	1.49e+06
native-country_Philippines	1.2860	7.61e+05	1.69e-06	1.000	-1.49e+06	1.49e+06
native-country_Poland	1.0182	7.61e+05	1.34e-06	1.000	-1.49e+06	1.49e+06
native-country_Portugal	1.0426	7.61e+05	1.37e-06	1.000	-1.49e+06	1.49e+06
native-country_Puerto-Rico	0.7247	7.61e+05	9.52e-07	1.000	-1.49e+06	1.49e+06
native-country_Scotland	0.7572	7.61e+05	9.94e-07	1.000	-1.49e+06	1.49e+06
native-country_South	-0.2820	7.61e+05	-3.7e-07	1.000	-1.49e+06	1.49e+06
native-country_Taiwan	0.7804	7.61e+05	1.02e-06	1.000	-1.49e+06	1.49e+06
native-country_Thailand	0.3334	7.61e+05	4.38e-07	1.000	-1.49e+06	1.49e+06
native-country_Trinadad&Tobago	0.5839	7.61e+05	7.67e-07	1.000	-1.49e+06	1.49e+06
native-country_United-States	1.2093	7.61e+05	1.59e-06	1.000	-1.49e+06	1.49e+06
native-country_Vietnam	-0.2307	7.61e+05	-3.03e-07	1.000	-1.49e+06	1.49e+06
native-country_Yugoslavia	1.7033	7.61e+05	2.24e-06	1.000	-1.49e+06	1.49e+06

==========

下面是一些关键的观察：

1. 模型拟合质量：

伪 R 平方为 0.4244，表示模型对观测变量的变异性有较好的解释能力，但仍有一定的未解释变异。

2. 数值稳定性问题：

部分系数（coef）的标准误差为 NaN，可能是因为数值不稳定导致的。这可能是由于变量之间存在共线性或其他数值问题。

3. 年龄（age）：

年龄的系数为正（0.0255），表明年龄的增加与更高的收入水平相关。这与通常的观察一致，即随着年龄的增长，人们的工作经验和职业地位也可能提高。

4. 受教育程度（education-num）：

教育程度的系数没有提供标准误差，但对于其他受教育程度的系数为正，表明更高的受教育程度可能与更高的收入水平相关。

5. 职业（occupation）：

不同职业的系数对收入的影响有所不同。例如，执行管理职业（Exec-managerial）和保护服务职业（Protective-serv）对应正系数，可能与高收入相关，而农业和渔业职业（Farming-fishing）对应负系数，可能与较低的收入相关。

6. 家庭状况 (marital-status):

不同婚姻状况的系数也有所不同。已分居 (Separated) 和从未结过婚 (Never-married) 的系数为负，可能与较低的收入水平相关。

7. 国籍 (native-country):

不同国家的系数差异较大，例如加拿大 (Canada) 和美国 (United-States) 的系数为正，可能与较高的收入水平相关，而荷兰（Holand-Netherlands）的系数为负，可能与较低的收入水平相关。

8. 性别 (sex):

性别的系数表明，女性（Female）和男性（Male）的系数都为负，但具体的影响需要进一步分析。可能存在性别差异对收入的影响。

9. 资本收益 (capital-gain) 和资本损失 (capital-loss):

资本收益和资本损失的系数都为正，表明这两个变量可能与更高的收入水平相关。

下面是变量显著性情况。

显著的变量:

1. Age (年龄): $P < 0.05$
2. fnlwgt: $P < 0.05$
3. Capital-gain (资本收益): $P < 0.05$
4. Capital-loss (资本损失): $P < 0.05$
5. Hours-per-week (每周工作小时): $P < 0.05$
6. Workclass（工作类别）中的一些类别
7. Education（教育程度）中的一些类别
8. Occupation（职业）中的一些类别
9. Marital-status（婚姻状况）中的一些类别
10. Native-country（国籍）中的一些类别

不显著的变量:

1. Workclass_State-gov
2. Workclass_Without-pay
3. Education_10th
4. Education_11th
5. Education_12th
6. Education_Assoc-acdm
7. Education_Assoc-voc
8. Education_Bachelors
9. Education_Doctorate
10. Education_HS-grad
11. Education_Masters
12. Education_Preschool
13. Education_Prof-school

14. Education_Some-college
15. Marital-status_Divorced
16. Marital-status_Married-AF-spouse
17. Marital-status_Married-civ-spouse
18. Marital-status_Married-spouse-absent
19. Marital-status_Never-married
20. Marital-status_Separated
21. Marital-status_Widowed
22. Occupation_Adm-clerical
23. Occupation_Armed-Forces
24. Occupation_Craft-repair
25. Occupation_Exec-managerial
26. Occupation_Farming-fishing
27. Occupation_Handlers-cleaners
28. Occupation_Machine-op-inspct
29. Occupation_Other-service
30. Occupation_Priv-house-serv
31. Occupation_Prof-specialty
32. Occupation_Protective-serv
33. Occupation_Sales
34. Occupation_Tech-support
35. Occupation_Transport-moving
36. Relationship_Husband
37. Relationship_Not-in-family
38. Relationship_Other-relative
39. Relationship_Own-child
40. Relationship_Unmarried
41. Relationship_Wife
42. Race_Amer-Indian-Eskimo
43. Race_Asian-Pac-Islander
44. Race_Black
45. Race_Other
46. Race_White
47. Sex_Female
48. Sex_Male
49. Native-country_Cambodia
50. Native-country_Canada
51. Native-country_China
52. Native-country_Columbia
53. Native-country_Cuba
54. Native-country_Dominican-Republic
55. Native-country_Ecuador
56. Native-country_El-Salvador
57. Native-country_England

58. Native-country_France
59. Native-country_Germany
60. Native-country_Greece
61. Native-country_Guatemala
62. Native-country_Haiti
63. Native-country_Holand-Netherlands
64. Native-country_Honduras
65. Native-country_Hong
66. Native-country_Hungary
67. Native-country_India
68. Native-country_Iran
69. Native-country_Ireland
70. Native-country_Italy
71. Native-country_Jamaica
72. Native-country_Japan
73. Native-country_Laos
74. Native-country_Mexico
75. Native-country_Nicaragua
76. Native-country_Outlying-US(Guam-USVI-etc)
77. Native-country_Peru
78. Native-country_Philippines
79. Native-country_Poland
80. Native-country_Portugal
81. Native-country_Puerto-Rico
82. Native-country_Scotland
83. Native-country_South
84. Native-country_Taiwan
85. Native-country_Thailand
86. Native-country_Trinadad&Tobago
87. Native-country_United-States
88. Native-country_Vietnam
89. Native-country_Yugoslavia

六、 其余指标

实验在测试集上计算了 F1 分数和 ROC 曲线下面积（ROC AUC Score），得到了以下结果：F1 Score: 0.6694395796847636、ROC AUC Score: 0.9070324839696606。

```
ROC AUC Score: 0.9070324839696606
Accuracy: 0.8498110204893574
Recall: 0.6109483217900906
F1 Score: 0.6694395796847636
```

分析如下：

1. F1 分数 (F1 Score) = 0.6694:

F1 分数相对较高，说明模型在精确性和召回率之间取得了良好的平衡。对于工资预测这样的二元分类任务，高 F1 分数表示模型在准确地预测正例和负例方面表现较好。可能存在一定的类别不平衡，而 F1 分数对不平衡数据具有较好的鲁棒性。

2. ROC 曲线下面积 (ROC AUC Score) = 0.9070:

高 ROC AUC Score 说明模型的性能在不同阈值下都很好。ROC AUC Score 接近于 1，表示模型在真正例率和假正例率之间的平衡非常良好。在工资预测中，这可能意味着模型对于不同工资水平的预测都能够保持较高的准确性。

结合这两个指标，可以得出模型在工资预测任务中表现出色。然而，具体的评估也应该考虑业务背景和任务需求。例如，如果在工资预测中对高工资的预测更为关键，那么可能需要进一步关注高工资水平的预测性能。

七、 改进模型

7.1 改进方法

作者根据《智能计算系统》课程和《人工智能数学基础》课程所学，决定对模型进行正则化处理。

代码如下:

```
def train_model(X, Y, regularization='l1', alpha=1.0):
    X = sm.add_constant(X)
    if regularization.lower() == 'l1':
        print("use-l1")
        model = sm.Logit(Y, X)
        result_model = model.fit_regularized(method='l1', alpha=alpha)
    elif regularization.lower() == 'l2':
        print("use-l2")
        model = sm.Logit(Y, X)
        result_model = model.fit_regularized(method='l2', alpha=alpha)
    else:
        raise ValueError("Invalid regularization type. Use 'l1' or 'l2'.")
    return result_model
```

代码意义:

1.数据准备:

函数接受特征矩阵 X 和目标变量 Y 作为输入。

使用 sm.add_constant(X) 将特征矩阵 X 添加截距项，以便逻辑回归模型可以学习截距。

2.正则化选择:

函数通过 regularization 参数来选择正则化类型，支持 'l1' 和 'l2' 两种正则化。

alpha 参数控制正则化的强度。较大的 alpha 值通常会导致更强的正则化。

3.模型训练：

使用 Statsmodels 的 Logit 类构建逻辑回归模型。

通过 fit_regularized 方法进行模型拟合。选择的正则化方法（L1 或 L2）由 method 参数指定，而正则化强度由 alpha 参数控制。

4.返回结果：

函数返回拟合的逻辑回归模型。

5.打印信息：

在函数中添加了一些用于输出信息的 print 语句，以指示使用了哪种正则化方法。

6.异常处理：

函数对正则化类型进行了验证，如果选择了无效的正则化类型，将引发 ValueError。

7.2 改进原理

1. L1 正则化（Lasso 正则化）：

- **目标函数：**
  在逻辑回归中，L1 正则化的目标函数由以下两部分组成：
  $J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y_i \log(h_\theta(x_i)) + (1-y_i)\log(1-h_\theta(x_i))] + \alpha\sum_{j=1}^{n}|\theta_j|$
- **作用：**
  L1 正则化通过添加 $\alpha\sum_{j=1}^{n}|\theta_j|$ 项来限制模型参数的绝对值。这鼓励模型参数稀疏化，即某些参数趋向于几乎为零，这使得模型更容易解释，并且可以在某种程度上自动进行特征选择。
- **影响：**
  - 通过将某些特征的系数推到零，L1 正则化有助于对不相关或冗余的特征进行特征选择，减少过拟合的风险。
  - L1 正则化在某些情况下也可以提高模型的泛化性能。

2. L2 正则化（Ridge 正则化）：

- **目标函数：**
  L2 正则化的目标函数由以下两部分组成：
  $J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y_i \log(h_\theta(x_i)) + (1-y_i)\log(1-h_\theta(x_i))] + \alpha\sum_{j=1}^{n}\theta_j^2$
- **作用：**
  L2 正则化通过添加 $\alpha\sum_{j=1}^{n}\theta_j^2$ 项来限制模型参数的平方和。这有助于防止模型的参数过大，从而减轻了过拟合的可能性。
- **影响：**
  - L2 正则化有助于处理共线性（特征之间存在高度相关性）的问题，使模型对特征中的小变化更加稳健。
  - L2 正则化在某些情况下可以提高模型的泛化性能。

参数解释：

- α 是正则化强度的超参数，控制正则化的力度。较大的 α 会增加正则化的效果，促使模型更加简单。
- θ 是模型的参数。
- $h_\theta(x)$ 是逻辑回归的假设函数。
- $J(\theta)$ 是目标函数，包括逻辑回归的损失函数和正则化项。

7.3 改进效果

原模型性能：

```
ROC AUC Score: 0.9070324839696606
Accuracy: 0.8498110204893574
Recall: 0.6109483217900906
F1 Score: 0.6694395796847636
Cross-Validation Accuracy: 0.790763224897898
Cross-Validation recall: 0.26558423988978497
```

改进后模型性能：（use-l1）

```
ROC AUC Score: 0.9033139932704346
Accuracy: 0.7598935226264418
Recall: 0.6075380359612724
F1 Score: 0.7220053420998562
```

可见模型性能有所提高。