1．实验目的

（1）掌握Pandas Dataframe的使用方法。

（2）掌握数据探索基本方法。

（3）掌握数据可视化基本方法。

（4）能够处理缺失值、与重复值。

（5）能够选择合适的数值规范化方法。

2．任务描述

2.1 金融欺诈数据的特征工程

请综合运用你所掌握的数据预处理方案，清洗数据，筛选重要的特征，或者对数据特征做合理变换，尽量提高逻辑回归分类器的性能，看看自己可以将准确率提高到多少。

2.2 数据集

银行贷款数据，一共包含690个样本，16个属性。基于隐私保护的考虑数据已经脱敏，属性名称用A1，A2，…，A16来表示，其中A16是类标号属性，有两个取值：“+”和“﹣”，“+”表示信贷申请被通过，“﹣”表示信贷申请被否决。需要注意的是，一些属性的值为“NA”，表示值缺失。

3．相关知识

数据预处理的主要步骤包括：

（1）数据清洗：这是数据预处理的首要任务。数据清洗涉及处理数据中的噪声、错误和异常值。常见的清洗操作包括去除重复记录、纠正错误数据、处理异常值等。

（2）处理缺失值：在现实世界的数据中，缺失值是很常见的。预处理过程需要对缺失值进行处理，常见的方法包括删除含有缺失值的记录、用平均值或中位数填充缺失值、使用插值方法进行填充等。

（3）数据转换：在许多情况下，原始数据并不适合直接用于分析和建模，需要进行数据转换。数据转换包括对数据进行归一化、标准化、对数化、幂变换等，以确保数据在相同尺度上进行比较和处理。

（4）数据集成：在现实应用中，数据通常来自不同的数据源，需要将多个数据源集成为一个完整的数据集。数据集成涉及到处理不同数据源之间的不一致性和冗余问题。

（5）特征选择：在拥有大量特征的数据集中，选择与任务相关且最具有代表性的特征对于提高模型性能至关重要。特征选择可以帮助减少模型的复杂性、降低维度和提高模型的泛化能力。

（6）数据降维：当数据集中包含大量特征时，降低数据的维度可以减少计算复杂性，并帮助可视化数据。主成分分析（PCA）和线性判别分析（LDA）是常见的数据降维技术。

数据预处理的质量直接影响后续分析和建模的结果，因此在进行数据预处理时应该谨慎且细致。预处理过程需要根据具体问题和数据的特点灵活选择合适的方法，以确保最终得到高质量的数据，从而为后续的任务打下坚实的基础。

4．实验要求与步骤

记录技术方案、优化思想，以及关键代码和中间步骤的运行结果。

1.data\_unique=data.drop\_duplicates() #去重

2. 处理缺失值，按最大概率填充

t = data\_ori.isnull().sum()

print(t[t>0])

data\_ori['A1'].value\_counts()

data\_ori['A1'].unique()

data\_ori['A1'].fillna('b', inplace=True)

1. 非数值型编码

#处理缺失值

t = data\_ori.isnull().sum()

print(t[t>0])

data\_ori['A1'].value\_counts()

data\_ori['A1'].unique()

data\_ori['A1'].fillna('b', inplace=True)

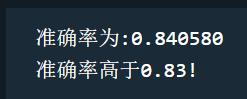
1. 规范化

    from sklearn.preprocessing import MinMaxScaler

    scaler = MinMaxScaler()

    X = scaler.fit\_transform(X)

5．测试说明



1. 分析与讨论\

（1）为什么需要去除数据集中的重复值？

因为重复值大概率是输入错误导致的

（2）填充缺失值可以采取哪些策略？

均值，中位数，众数，线性回归，决策树回归，随机森林回归，One-hot

（3）何时需要对数据做规范化？本实验数据，适合哪种规范化方法？

最大最小值差的很大。

1. 附录

import numpy as np

import pandas as pd

from sklearn.preprocessing import OrdinalEncoder

import warnings

warnings.filterwarnings('ignore')

data\_ori = pd.read\_csv('/data/bigfiles/bankloan.csv')

data\_ori.head()

data\_ori.info()

data\_ori.describe()

#处理缺失值

t = data\_ori.isnull().sum()

print(t[t>0])

data\_ori['A1'].value\_counts()

data\_ori['A1'].unique()

data\_ori['A1'].fillna('b', inplace=True)

data\_ori['A4'].value\_counts()

data\_ori['A4'].fillna('u',inplace=True)

data\_ori['A5'].value\_counts()

data\_ori['A5'].fillna('g',inplace=True)

data\_ori['A6'].value\_counts()

data\_ori['A6'].fillna('c',inplace=True)

data\_ori['A7'].value\_counts()

data\_ori['A7'].fillna('v',inplace=True)

#非数值字段编码

cols = [col for col in data\_ori.select\_dtypes(include=['object']).columns]

enc=OrdinalEncoder()

newx=enc.fit\_transform(data\_ori[cols])

newx[:5]

data\_ori[cols]=newx

print(data\_ori.iloc[:,:6].describe())