**泰坦尼克号乘客生存数据预处理报告**

**根据自己写的论文进行参考。**

**1. 数据预处理概述**

本报告详细描述了泰坦尼克号乘客生存数据集的预处理过程。预处理是数据分析的关键步骤，直接影响后续模型的性能。我们的预处理流程包括数据清洗、特征工程、特征编码和数据标准化等步骤。

**2. 原始数据描述**

原始数据包含以下特征：

* 乘客ID、姓名、性别、年龄
* 舱位等级、同船亲属数量、船票信息
* 票价、船舱号码、登船港口
* 生存状态(标签)

**3. 数据预处理步骤**

**3.1 数据清洗**

import pandas as pd

import numpy as np

from sklearn.preprocessing import StandardScaler

# 加载数据

train\_df = pd.read\_csv('train.csv')

test\_df = pd.read\_csv('test.csv')

# 处理缺失值

def handle\_missing\_values(df):

# 年龄用均值填充

df['Age'].fillna(df['Age'].mean(), inplace=True)

# 船舱号码用众数填充

df['Cabin'].fillna(df['Cabin'].mode()[0], inplace=True)

# 登船港口用众数填充

df['Embarked'].fillna(df['Embarked'].mode()[0], inplace=True)

return df

train\_df = handle\_missing\_values(train\_df)

test\_df = handle\_missing\_values(test\_df)

**3.2 特征工程**

# 特征工程函数

def feature\_engineering(df):

# 从姓名中提取称呼

df['Title'] = df['Name'].apply(lambda x: x.split(',')[1].split('.')[0].strip())

# 创建亲属总数特征

df['FamilySize'] = df['SibSp'] + df['Parch']

# 创建是否独自旅行特征

df['IsAlone'] = (df['FamilySize'] == 0).astype(int)

# 票价等级

df['FareLevel'] = (df['Fare'] > df['Fare'].mean()).astype(int)

# 年龄等级

df['AgeLevel'] = (df['Age'] > df['Age'].mean()).astype(int)

return df

train\_df = feature\_engineering(train\_df)

test\_df = feature\_engineering(test\_df)

**3.3 特征编码**

from sklearn.preprocessing import LabelEncoder

# 分类变量编码

def encode\_features(df):

# 性别编码

le = LabelEncoder()

df['Sex'] = le.fit\_transform(df['Sex'])

# 登船港口编码

df['Embarked'] = le.fit\_transform(df['Embarked'])

# 称呼编码

title\_mapping = {'Mr': 1, 'Miss': 2, 'Mrs': 3, 'Master': 4, 'Dr': 5,

'Rev': 6, 'Col': 7, 'Major': 8, 'Mlle': 9, 'Countess': 10,

'Ms': 11, 'Lady': 12, 'Jonkheer': 13, 'Don': 14, 'Dona': 15,

'Mme': 16, 'Capt': 17, 'Sir': 18}

df['Title'] = df['Title'].map(title\_mapping)

df['Title'].fillna(0, inplace=True)

return df

train\_df = encode\_features(train\_df)

test\_df = encode\_features(test\_df)

**3.4 数据标准化**

# 数值特征标准化

def standardize\_features(df):

scaler = StandardScaler()

numeric\_features = ['Age', 'Fare', 'SibSp', 'Parch', 'FamilySize']

df[numeric\_features] = scaler.fit\_transform(df[numeric\_features])

return df

train\_df = standardize\_features(train\_df)

test\_df = standardize\_features(test\_df)

**3.5 特征选择**

# 选择最终使用的特征

selected\_features = ['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare',

'Embarked', 'Title', 'FamilySize', 'IsAlone',

'FareLevel', 'AgeLevel']

X\_train = train\_df[selected\_features]

y\_train = train\_df['Survived']

X\_test = test\_df[selected\_features]

**4. 预处理结果分析**

经过预处理后，我们得到了以下特征：

1. 原始特征：舱位等级、性别、年龄、票价等
2. 衍生特征：亲属总数、是否独自旅行、票价等级等
3. 编码特征：性别、登船港口、称呼等

所有数值特征都已标准化，消除了量纲影响。

**5. 数据可视化代码示例**

import matplotlib.pyplot as plt

import seaborn as sns

# 生存率与性别的关系

plt.figure(figsize=(8, 5))

sns.barplot(x='Sex', y='Survived', data=train\_df)

plt.title('Survival Rate by Gender')

plt.xlabel('Gender (0=Male, 1=Female)')

plt.ylabel('Survival Rate')

plt.show()

# 生存率与舱位等级的关系

plt.figure(figsize=(8, 5))

sns.barplot(x='Pclass', y='Survived', data=train\_df)

plt.title('Survival Rate by Passenger Class')

plt.xlabel('Passenger Class')

plt.ylabel('Survival Rate')

plt.show()

# 年龄分布

plt.figure(figsize=(10, 6))

sns.histplot(data=train\_df, x='Age', hue='Survived', kde=True, bins=30)

plt.title('Age Distribution by Survival Status')

plt.xlabel('Age')

plt.ylabel('Count')

plt.show()

**6. 模型训练示例代码**

from sklearn.linear\_model import LogisticRegression

from sklearn.model\_selection import cross\_val\_score

from sklearn.metrics import classification\_report

# 初始化逻辑回归模型

logreg = LogisticRegression(max\_iter=1000)

# 交叉验证

cv\_scores = cross\_val\_score(logreg, X\_train, y\_train, cv=5, scoring='accuracy')

print(f"Cross-validation accuracy scores: {cv\_scores}")

print(f"Mean CV accuracy: {np.mean(cv\_scores):.4f}")

# 训练最终模型

logreg.fit(X\_train, y\_train)

# 预测训练集

y\_pred = logreg.predict(X\_train)

# 分类报告

print("\nClassification Report:")

print(classification\_report(y\_train, y\_pred))

**7. 结论**

通过系统的数据预处理，我们：

1. 处理了缺失值问题
2. 创建了有意义的衍生特征
3. 对分类变量进行了适当编码
4. 标准化了数值特征
5. 选择了最相关的特征集

这些预处理步骤为后续的建模分析奠定了良好基础，有助于提高模型的性能和解释性。