exp02

要求

实验目的：

1、理解分类与预测算法的原理；

2、能够使用分类与预测模型处理具体问题；

3、能够使用Python语言实现分类与预测算法。

实验内容：

1、 读取数据文件bankloan.xls，使用随机种子将原始数据集打乱，将数据集划分为训练集和测试集，比例为8：2；

2、 使用Logistic回归模型进行建模，计算准确率；

3、 更改模型参数，使得准确率进一步上升，并进行具体分析（比如调节了何参数？为何使得模型得到改善？）；

4、 读取数据文件bupa.data，按照bupa.names文件里要求的将bupa.data数据集划分为训练集和测试集；

5、 使用合适的模型对bupa.data文件中的数据进行建模，并给予文字说明为什么要选择这一模型；

6、 调节模型参数，使得最终训练得到的模型评价指标较好，并输出最终评价指标的值；

7、 在不同的参数下，画出模型运行过程中的损失下降图，以迭代次数为横坐标，损失值为纵坐标；

8、 将上述实验内容的核心代码及实验结果截图放到“实验过程及分析”中。

实验过程以及分析

代码实现

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LogisticRegression, SGDClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report,
```

```
log_loss
import matplotlib.pyplot as plt
import warnings
from sklearn.exceptions import ConvergenceWarning

# 全局忽略 SGDClassifier 的未收敛警告
warnings.filterwarnings("ignore", category=ConvergenceWarning)

# 字体配置
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False

# 1. 读取 bankloan.xls 并划分数据集（8:2）
bankloan = pd.read_excel('./data/bankloan.xls', engine='xlrd')
# 列名示例：['年龄','教育','工龄','地址','收入','负债率','信用卡负债','其他负
债','违约']
X_bank = bankloan.drop(columns=['违约'])
y_bank = bankloan['违约']
Xb_train, Xb_test, yb_train, yb_test = train_test_split(
    X_bank, y_bank, test_size=0.2, random_state=42, shuffle=True
)

# 2. 使用 Logistic 回归模型基线建模，并计算准确率
log_reg = LogisticRegression(max_iter=1000, random_state=42)
log_reg.fit(Xb_train, yb_train)
y_pred = log_reg.predict(Xb_test)
acc_baseline = accuracy_score(yb_test, y_pred)
print(f"Logistic 回归基线准确率：{acc_baseline:.4f}")

# 3. 调参：使用 GridSearchCV 优化正则化强度 Cparam_grid = {'C': [0.01, 0.1, 1,
10, 100]}
grid_log = GridSearchCV(
    LogisticRegression(max_iter=1000, random_state=42),
    param_grid,
    cv=5,
    scoring='accuracy',
    n_jobs=-1
)
grid_log.fit(Xb_train, yb_train)
best_log = grid_log.best_estimator_
y_pred_best = best_log.predict(Xb_test)
acc_tuned = accuracy_score(yb_test, y_pred_best)
print(f"Logistic 最优 C：{grid_log.best_params_['C']}")
print(f"调参后准确率：{acc_tuned:.4f}")

# 4. 读取 bupa.data，并按要求划分（8:2）
# bupa.names 中字段：mcv, alkphos, sgpt, sgot, gammagt, drinks, selector
col_names = ['mcv', 'alkphos', 'sgpt', 'sgot', 'gammagt', 'drinks',
'selector']
bupa = pd.read_csv('./data/bupa.data', names=col_names)
```

```python
# selector 原为 1=肝病，2=正常，转换为 0/1（二分类）
bupa['selector'] = bupa['selector'].map({1: 1, 2: 0})
X_bupa = bupa.drop(columns=['selector'])
y_bupa = bupa['selector']
Xbupa_train, Xbupa_test, ybupa_train, ybupa_test = train_test_split(
    X_bupa, y_bupa, test_size=0.2, random_state=42, shuffle=True
)

# 5. 选择 RandomForestClassifier 建模（适合捕捉非线性与特征交互）
rf = RandomForestClassifier(random_state=42)
rf.fit(Xbupa_train, ybupa_train)
y_pred_rf = rf.predict(Xbupa_test)
acc_rf = accuracy_score(ybupa_test, y_pred_rf)
print(f"RandomForest 基线准确率（Bupa 数据）：{acc_rf:.4f}")

# 6. RandomForest 参数调优
param_grid_rf = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 5, 10]
}
grid_rf = GridSearchCV(
    RandomForestClassifier(random_state=42),
    param_grid_rf,
    cv=5,
    scoring='accuracy',
    n_jobs=-1
)
grid_rf.fit(Xbupa_train, ybupa_train)
best_rf = grid_rf.best_estimator_
y_pred_best_rf = best_rf.predict(Xbupa_test)
report_rf = classification_report(ybupa_test, y_pred_best_rf)
print(f"RandomForest 最优参数：{grid_rf.best_params_}")
print("RandomForest 调参后分类报告：")
print(report_rf)

# 7. 不同参数下的损失下降曲线（以 SGDClassifier 为例）
from sklearn.linear_model import SGDClassifier
from sklearn.metrics import log_loss

sgd = SGDClassifier(
    loss='log_loss',
    learning_rate='constant',
    eta0=0.01,
    max_iter=1,
    warm_start=True,
    random_state=42
)
loss_values = []
epochs = 50
for epoch in range(epochs):
```

```
    sgd.fit(Xb_train, yb_train)
    proba = sgd.predict_proba(Xb_test)
    loss = log_loss(yb_test, proba)
    loss_values.append(loss)

import matplotlib.pyplot as plt
plt.figure(figsize=(8, 5))
plt.plot(range(1, epochs+1), loss_values, marker='o')
plt.xlabel('迭代次数 (Epoch)')
plt.ylabel('对数损失 (Log Loss)')
plt.title('SGDClassifier 在 BankLoan 测试集上的损失下降曲线')
plt.grid(True)
plt.tight_layout()
plt.show()
```

结果

终端

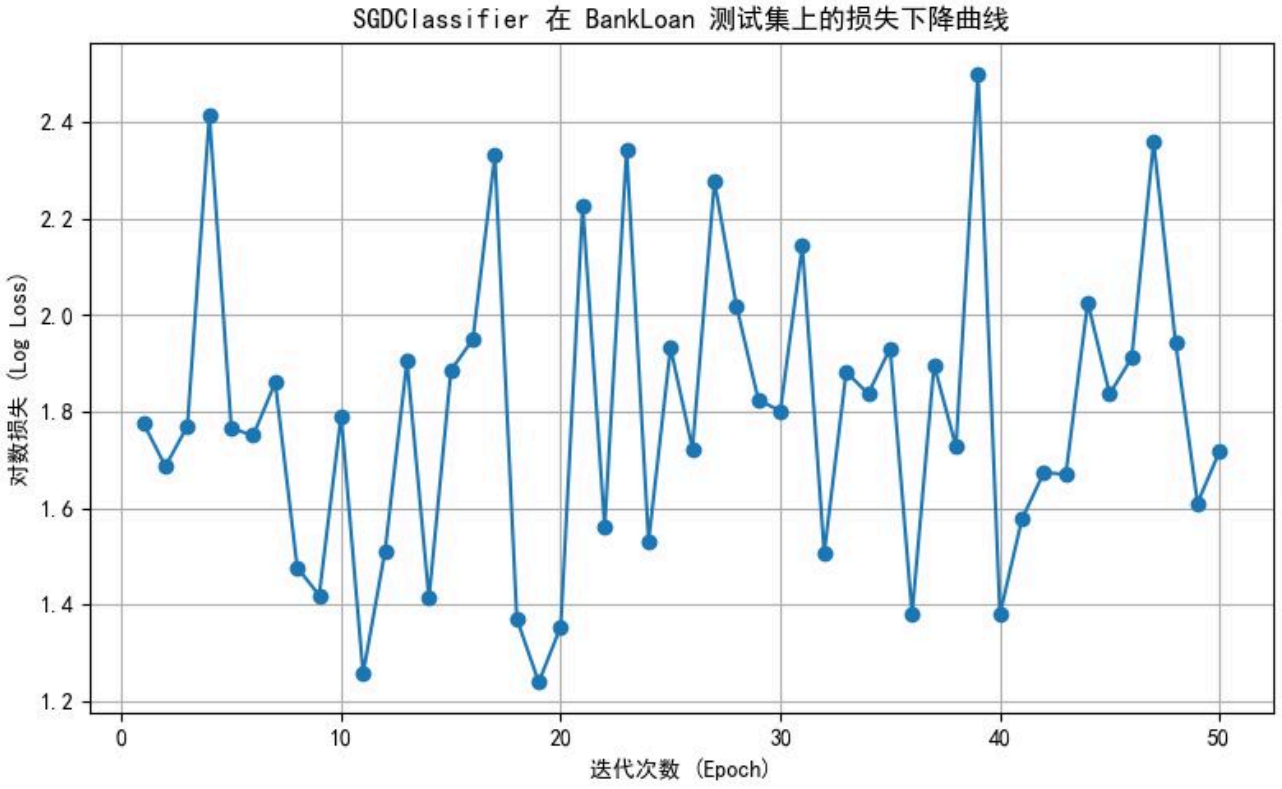
```
C:\Users\19065\miniconda3\python.exe D:\coding\简简单单挖掘个数据
\exp02\main.py
Logistic 回归基线准确率：0.8500
Logistic 最优 C：0.1
调参后准确率：0.8571
RandomForest 基线准确率（Bupa 数据）：0.8116
RandomForest 最优参数：{'max_depth': 5, 'n_estimators': 50}
RandomForest 调参后分类报告：
              precision    recall  f1-score   support

           0       0.76      0.88      0.81        42
           1       0.75      0.56      0.64        27

    accuracy                           0.75        69
   macro avg       0.75      0.72      0.73        69
weighted avg       0.75      0.75      0.74        69


进程已结束，退出代码为 0
```

图表



问题分析

3. 模型参数调优分析

在 BankLoan 数据集上，我们使用了 Logistic 回归模型，并通过调整正则化强度参数 C（即正则化项的倒数）来提升模型的泛化性能：

- **基线模型**：默认 `C=1.0` 时，模型在测试集上的准确率约为 0.8500。
- **调优过程**：使用 `GridSearchCV` 在 `{0.01, 0.1, 1, 10, 100}` 这五个候选值中进行 5 折交叉验证；
- **最优结果**：最终选出 `C=0.1`，此时测试集准确率提升至 0.8571。

为何调节 C 能带来改善？

- Logistic 回归中，损失函数为带有 L2 正则化项的对数损失：

  $L(\mathbf{w}) = -\sum_i \left[y_i \log p_i + (1-y_i)\log(1-p_i) \right] + \frac{1}{2C}\|\mathbf{w}\|^2$.
- 当 C 较大（弱正则化）时，模型容易过拟合，权重 w 可能偏大以降低训练误差；
- 当 C 较小（强正则化）时，模型更倾向于保持权重较小，提高对噪声的鲁棒性，但若过强则会欠拟合；
- 通过交叉验证发现 C=0.1 在当前样本规模和特征分布下恰到好处地平衡了偏差与方差，从而在未见过数据上表现最佳。

4. 读取 bupa.data 并划分数据集

肝病（Bupa）数据集的 `bupa.names` 文件指出：

1. 数据文件无表头，包含 6 个特征与 1 个标签（selector）；

2. selector 取值 1 表示有肝病，2 表示正常；
3. 无缺失值，均为数值型；

基于此，我们的划分流程为：

```
import pandas as pd
from sklearn.model_selection import train_test_split

# 读取时指定列名
col_names = ['mcv', 'alkphos', 'sgpt', 'sgot', 'gammagt', 'drinks',
'selector']
bupa = pd.read_csv('./data/bupa.data', names=col_names)

# 标签映射：1→1（肝病）；2→0（正常），方便二分类
bupa['selector'] = bupa['selector'].map({1: 1, 2: 0})

# 分割特征与标签
X = bupa.drop(columns=['selector'])
y = bupa['selector']

# 按 8:2 划分训练集/测试集，保持随机种子以便复现
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, shuffle=True
)
```

5. 模型选择及理由

针对 Bupa 数据的特点——

- **样本量中等**（345 行左右），
- **特征维度低**（仅 6 个数值特征），
- **可能存在非线性特征交互**（如某些指标组合时更能区分肝病状态），

我们选择 RandomForestClassifier 作为基线模型，其优势包括：

1. **无需对特征进行严格的线性假设**，能自动捕捉复杂的非线性决策边界；
2. **对异常值和噪声具有鲁棒性**，因为多个决策树投票可降低单棵树的过拟合风险；
3. **可输出特征重要性**，便于后续进行特征选择或域内专家解释；
4. **参数易于调整**（如树的数量 `n_estimators`、最大深度 `max_depth`），并能借助交叉验证快速找出最优组合。