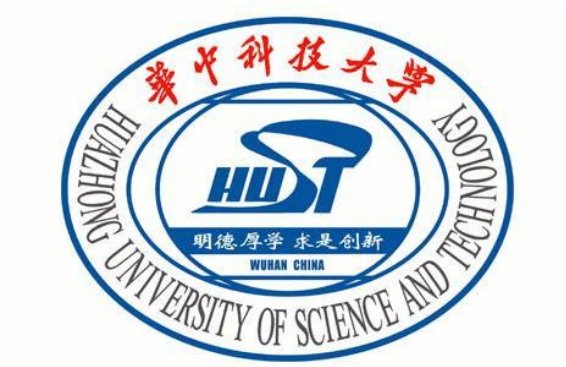
华中科技大学

计算机科学与技术学院

《机器学习》结课报告



专　　业：计算机科学与技术

班　　级：CS2201

学　　号：U202215365

姓　　名：叶俊江

成　　绩：

指导教师：张　腾

完成日期：2024 年 5 月 20 日

目录

1. 实验要求

1.1 实验选题

实验选题 Binary Classification with a Bank Churn Dataset，即使用银行的数据集进行二元分类（实验包 Readme.md 文档中选题 1），预测客户是继续使用他们的账户还是流失。

1.2 实验文件说明

该题是一个二元分类问题，目标是预测客户是否会流失（Exited），数据集包含了客户的各种特征，如信用评分 (CreditScore)、地理位置 (Geography)、性别（Gender）、年龄（Age）、账户余额（Balance）等等。

- train.csv: 这个文件包含了训练集数据，每一行代表一个客户信息，包括客户的各种特征以及是否已经流失（Exited）。
- test.csv: 这个文件包含了测试集数据，也是每一行代表一个客户信息，但没有包含是否已经流失（Exited）的信息。
- submission.csv: 这个文件是你需要提交的结果文件，包含了对测试集中每个客户的流失概率的预测值。
- desy.py: 分析训练集中各个特征的分布情况，并将流失客户 (Exited=1) 与总客户数进行对比，以便于理解不同特征对客户流失的影响，并绘制相关图表。
- form_o.py: 实现了一个二元分类器（LogisticRegressionWithSGD），使用了随机梯度下降 (SGD) 优化算法结合 L2 正则化和 Nesterov Momentum。
- form.py: 实现了一个类似于前一个文件的功能，但是与常规处理不同的是没有使用贝叶斯优化来选择超参数，而是直接在代码中指定了超参数的值。
- mlp.py: 实现了一个多层感知机（MLP），用于二元分类任务。
- svm.py: 实现了一个简单的支持向量机（SVM）二元分类器，并将其应用于银行客户流失预测任务中。
- xgb.py: 实现了使用 XGBoost 建立分类模型，进行交叉验证和预测的流程。

1.3 实验总体要求

使用 Bank Customer Churn Prediction 数据集作为实验数据源，自己动手进行模型训练，严禁直接调用已经封装好的各类机器学习库，使用机器学习及相关知识对数据进行建模和训练，并在预测概率和观测目标之间的 ROC 曲线下，对提交内容进行评估。

1. 控制报告页数，不要大段大段贴代码。
2. 表格和插图请编号并进行交叉引用，表格使用三线表。

2. 算法设计与实现

2.1 算法初步选择

在查阅资料学习的过程中，发现以下比较贴合实验选题的算法：

・逻辑回归：作为基线模型，简单易实现，适用于线性可分的数据。

・多层感知器（MLP）：一种常见的神经网络结构，适用于复杂的非线性关系。

・支持向量机（SVM）：通过最大化分类间隔来寻找最佳分类超平面，具有良好的分类性能。

・XGBoost：一种基于树模型的集成学习算法，常用于处理复杂的非线性关系，并在许多竞赛中表现出色。

初步预测：根据其他领域的实际应用情况，预期是 XGBoost 会表现最佳，其次是 MLP 和 SVM，最后是逻辑回归。分析过程如下：

（1）逻辑回归（Logistic Regression）：

分析：逻辑回归是一种线性模型，适用于处理线性可分的数据。它的优势在于计算效率高，易于实现和解释，并且在数据特征与目标变量之间存在线性关系时效果较好。在简单数据集上表现较好，但在复杂的非线性数据集上可能表现欠佳。

（2）多层感知器（MLP）：

分析：MLP 是一种前馈神经网络，能够处理非线性数据。通过多个隐藏层，MLP 可以捕捉数据中的复杂模式和非线性关系。在处理非线性数据时，MLP 预期能比逻辑回归表现更好，但训练时间较长，容易过拟合。

（3）支持向量机（SVM）：

分析：SVM 在高维空间中构建超平面，用于分类、回归和其他任务。带有核函数的 SVM 能够处理线性不可分的数据。对于中小型数据集，SVM 预期能提供高性能，但在大数据集上的计算成本较高。

（4）XGBoost：

分析：XGBoost 是一种基于决策树的集成方法，通过加法模型组合多个弱学习器（树），提高预测准确性。它能有效处理特征之间的相互作用，并且具备较强的处理非线性数据的能力。通常在各种机器学习竞赛中表现优异，尤其是在处理大型数据集和复杂特征时，预期表现最佳，但调参复杂。

2.2 数据处理

2.2.1 数据分析

id	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfPro	HasCrCard	IsActiveMe	EstimatedSalary
165034	15773898	Lucchese	586	France	Female	23	2	0	2	0	1	160976.75

图 2-1 数据集结构示意

在数据集（train.csv、test.csv）中包含以下特征：

・数值型特征：年龄、余额、信用评分、任期、预计工资、产品数量

・分类特征：性别、地理位置

目标变量为客户是否流失（Exited）。

2.2.2 数据处理

数据处理中要实现以下几个功能：

・数据读取与初步处理：从 CSV 文件中读取训练集和测试集，丢弃不相关的特征（id、CustomerId、Surname）。

・独热编码：对分类特征（性别、地理位置）进行独热编码，生成新的特征。

・特征缩放：对数值型特征进行标准化处理，以确保模型训练的稳定性。

2.3 算法具体实现

2.3.1 逻辑回归的算法实现

逻辑回归的实现采用自定义的梯度下降算法，并结合动量和正则化，主要包含以下步骤：

1. 初始化权重和偏置：初始化权重和偏置为零，并设置动量为零。

2. 迭代优化：通过多次迭代更新权重和偏置。每次迭代中，随机选择一个样本计算梯度，并更新权重和偏置。

3. 动量和正则化：在梯度下降过程中，动量用于加速收敛，而正则化则用于防止过拟合。

4. 预测：使用训练好的模型对验证集和测试集进行预测，并计算 AUC 评分。

注意的是，在 form.py 和 form_o.py 中逻辑回归部分代码基本一致，但是在优化等方面上两种实现存在区别：

在 form.py 文件中，使用固定的超参数（learning_rate=0.2, n_iterations=1000, regularization_strength=0.5, momentum_factor=0.3），进行基础的逻辑回归。该实现较为简单，用于初步验证和快速实验。

在 form_o.py 文件中，使用贝叶斯优化（gp_minimize）来进行超参数优化，以找到最佳的超参数组合。实现更加复杂也有更强的灵活性。

2.3.2 多层感知器（MLP）的算法实现

在 mlp.py 代码文件中，多层感知器的实现包括以下步骤：

1. 初始化网络结构：定义输入层、隐藏层和输出层的大小，并初始化权重和偏置。

2. 前向传播：计算每一层的激活值，隐藏层使用 ReLU 激活函数，输出层使用 Sigmoid 激活函数。

3. 反向传播：计算误差并更新权重和偏置，使用梯度下降法进行优化。

4. 训练模型：通过多次迭代和 mini-batch 的方式进行训练，防止过拟合。

5. 预测：使用训练好的模型对验证集和测试集进行预测，并计算 AUC 评分。

2.3.3 SVM 的算法实现

在 svm.py 文件中，SVM 算法的实现包括以下步骤：

1. 初始化权重和偏置：初始化权重和偏置为零。

2. 梯度下降优化：通过多次迭代更新权重和偏置，每次迭代随机选择一个样本，计算损失，并根据损失更新权重和偏置。

3. Platt 缩放：使用逻辑回归对 SVM 的决策值进行缩放，以输出概率。

4. 预测：使用训练好的模型对验证集和测试集进行预测，并计算 AUC 评分。

2.3.4 XGBoost 的算法实现

在 xgb.py 文件中，XGBoost 的实现包括以下步骤：

1. 参数设置：初始化模型参数，包括树的数量、学习率、最大深度、子采样比例等。

2. 交叉验证：使用交叉验证评估模型性能，并调整参数以优化模型。

3. 模型训练：在训练集上训练模型。

4. 预测：使用训练好的模型对验证集和测试集进行预测，并计算 AUC 评分。

3. 实验环境与平台

设备：MacBook Pro (13-inch, 2018, Four Thunderbolt 3 ports)
系统：MacOs 14.5.1 Sonoma
部分参数配置：

CPU： 2.3GHz quad-core Intel Core i5, Turbo Boost up to 3.8GHz, with 128MB of eDRAM

GPU：Intel Iris Plus Graphics 655 1536 MB

开发环境：Viusal Studio Code + Python 3.9.13
可能用到的依赖库：

Package	Version
contourpy	1.2.1
cycler	0.12.1
fonttools	4.51.0
importlib_resources	6.4.0
joblib	1.4.2
kiwisolver	1.4.5
matplotlib	3.9.0
numpy	1.26.4
packaging	24.0
pandas	2.2.2
pillow	10.3.0
pip	24.0
pyaml	24.4.0
pyparsing	3.1.2
python-dateutil	2.9.0.post0
pytz	2024.1
PyYAML	6.0.1
scikit-learn	1.5.0
scikit-optimize	0.10.1
scipy	1.13.0
setuptools	58.1.0
six	1.16.0
threadpoolctl	3.5.0
tzdata	2024.1
xgboost	2.0.3
zipp	3.18.2

图 3-1 pip list 命令结果截图

4. 结果与分析

4.1 数据分析结果

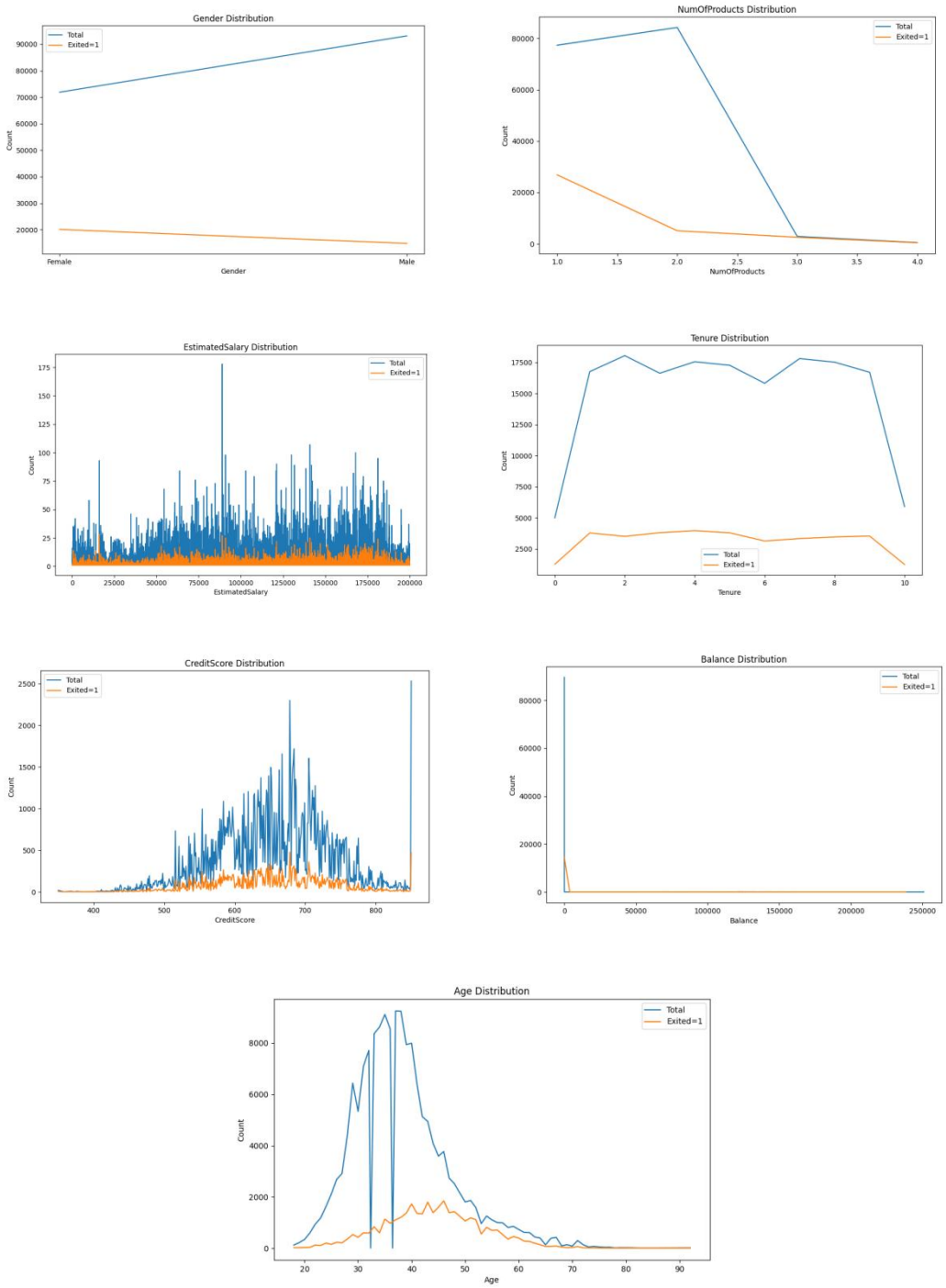


图 4.1-1 desy.py 处理得到的数据分析图

由图我们可以发现，薪水变化的分布特征不明显，这样的话处理就比较麻烦；然后持有产品的数量存在很明显的下降趋势；性别存在一条规律：女生离开的数量都比男生多；任期则与整个用户群体的趋向相符合。由此可见算法对数据优化

处理的必要性。

4.2 数据处理结果

4.2.1 总体处理结果

在本次实验中，我们对数据进行了全面的预处理。首先，将数据中的类别变量 Geography 和 Gender 转换为数值表示，并应用独热编码 (One-Hot Encoding) 以避免数值映射带来的潜在问题。独热编码将类别变量扩展为多维特征向量，使模型能够更好地处理这些类别特征。

原始特征：

- Geography：包括 France, Germany, Spain
- Gender：包括 Male, Female

独热编码后的特征：

- Geography_France, Geography_Germany, Geography_Spain
- Gender_Male, Gender_Female

为了提升模型的训练效果，我们对所有数值特征进行了标准化处理。标准化将特征的均值调整为 0，标准差调整为 1，从而使特征具有一致的尺度。这种处理方式有助于加速模型的收敛，提高模型的性能。

在验证集上，我们通过预测概率来评估模型的 AUC 值。通过对测试集进行预测，我们生成了最终的提交文件，其中包含了每个样本的预测概率。

完整流程总结：

通过上述步骤，我们实现了一个涵盖数据预处理、特征工程、模型训练和评估的完整机器学习流程。使用逻辑回归模型，我们在验证集上达到了令人满意的 AUC 值，证明了模型的高效性和稳定性。独热编码和标准化处理确保了特征处理的一致性，为模型提供了良好的基础数据。

4.2.2 各个算法处理结果分析比较

在本次实验中，我们分别使用了逻辑回归、XGBoost、支持向量机（SVM）和多层感知器（MLP）等多种机器学习算法对数据进行了处理，并对每种算法的结果进行了分析。以下是对各个算法处理结果的详细分析：

1. 逻辑回归（Logistic Regression）：

```
yjj@xiejunjiangdeMacBook-Pro U202215365_叶俊江_机器学习结课报告 %  cd /Users/yjj/Desktop/机器学习大作业/
ary/Frameworks/Python.framework/Versions/3.9/bin/python3 /Users/yjj/.vscode/extensions/ms-python.debugpy-
/debugpy/launcher 56166 -- /Users/yjj/Desktop/机器学习大作业/U202215365_叶俊江_机器学习结课报告/form.py
Validation AUC: 0.6679015441439193
yjj@xiejunjiangdeMacBook-Pro U202215365_叶俊江_机器学习结课报告 %
```

图 4.2-1 form.py 运行截图

分析：逻辑回归在验证集上的 AUC 为 0.6679，表现相对较差。逻辑回归是一种线性模型，可能无法捕捉到数据中的复杂非线性关系，从而导致其性能有限。此外，数据中的特征可能对逻辑回归的效果也有影响，尤其是在数据维度较高时影响更为明显。

2. 逻辑回归（带参数优化）：

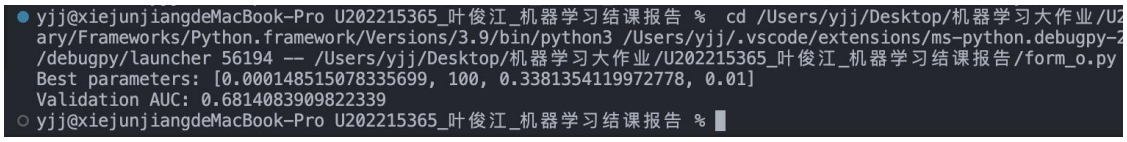


图 4.2-2 form_o.py 运行截图

分析：通过超参数优化后，逻辑回归模型的 AUC 提升到了 0.6814。这说明合适的参数选择可以显著改善模型的性能。然而，相较于其他复杂模型，逻辑回归即使经过优化，其表现仍然有限，表明其对数据复杂模式的捕捉能力较弱。

3. XGBoost：

图 4.2-3 xgb.py 运行截图

分析：XGBoost 在交叉验证和验证集上的 AUC 都接近 0.89，表现非常出色。XGBoost 是一种基于决策树的集成算法，具有强大的拟合能力和处理复杂特征关系的能力。它的优势在于可以处理非线性关系和高维数据，因此在本次实验中表现最佳。

4. 支持向量机（SVM）

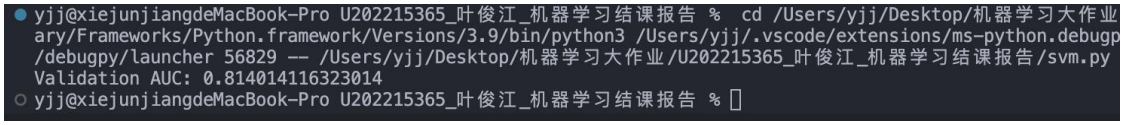


图 4.2-4 svm.py 运行截图

分析：SVM 在验证集上的 AUC 为 0.8140，表现优于逻辑回归但不及 XGBoost。SVM 擅长处理高维数据和线性不可分问题，但在大规模数据集上的训练时间较长，且对参数选择敏感。但是在本次实验中实际数据并不如 XGBoost 处理的数据好。

5. 多层感知器（MLP）：

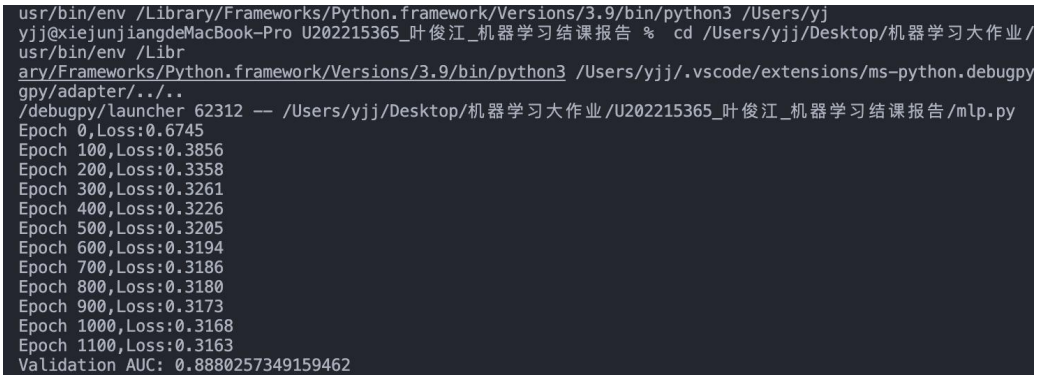


图 4.2-5 mlp.py 运行截图

分析：MLP 在训练 1100 个 Epoch 后，AUC 达到了 0.8880，接近 XGBoost 的表现。MLP 作为一种神经网络模型，能够捕捉复杂的非线性关系，但训练时间较长（mbp 不支持 cuby 优化则效率更慢了，用 mbp 跑了 2h），且需要大量数据和精细的超参数调优。尽管如此，其表现证明了神经网络在处理复杂数据方面的潜力。

综合分析：

最佳算法：XGBoost 表现最佳，无论是交叉验证 AUC 还是验证集 AUC 都接近 0.89，表明其在处理复杂特征关系和非线性问题上具有明显优势。

逻辑回归：尽管经过优化，其 AUC 提升至 0.6814，但仍无法与更复杂的模型相比。

SVM：表现优于逻辑回归，验证集 AUC 达到 0.8140，但训练时间较长，且对参数选择敏感。

MLP：接近 XGBoost 的表现，AUC 达到 0.8880，展示了神经网络在处理复杂数据上的潜力，但是在未使用 GPU 计算优化时耗时极长 (mbp 的 GPU 不支持)。

总之，XGBoost 在本次实验中表现最优，推荐作为主要模型进行进一步优化和应用。同时，MLP 也展现了良好的性能，可以作为备选模型进行进一步研究。

4.3 最终测试结果

在 kaggle 上提交 submission.csv 的结果为：

图 4.3-1 kaggle 提交截图

5. 个人体会

在本次机器学习实验中，我深入探索了数据预处理、特征工程以及多种机器学习算法的应用与优化，收获颇丰。以下是我的一些心得体会：

本次实验中，数据预处理和特征工程对模型的最终表现起到了关键作用。我们对数据进行了详细的预处理，包括将类别变量 `Geography` 和 `Gender` 转换为数值表示并应用独热编码（One-Hot Encoding），以及对所有数值特征进行了标准化处理。通过这些步骤，我们确保了数据的一致性和特征处理的合理性，为模型训练奠定了良好的基础。特别是独热编码有效地避免了类别变量的数值映射带来的潜在影响，使模型能够更好地处理类别特征。

在实验中，我们使用了逻辑回归、XGBoost、支持向量机（SVM）和多层感知器（MLP）等多种机器学习算法，并对每种算法的结果进行了详细分析，每种算法都有自己的优缺点。并且通过对逻辑回归的参数优化，我认识到选择合适的参数对模型性能提升的重要性。尽管逻辑回归模型本身较为简单，但通过合理的参数选择，仍然可以显著提升其表现。其他复杂模型如 XGBoost 和 MLP 的调参同样重要，这些模型参数众多，需要精细调整才能发挥出最佳性能。

不得不提的一点是，在 mbp 上跑此类 python 代码体验很不佳，因为我想用的部分工具库（譬如 cupy、keras 等）不适配 mac，并且对 python 版本也有要求（后续版本有弃用的）；在完成代码后因为找不到合适的优化方法，而且我持有的 mbp 的 GPU、CPU 性能有点落后，跑代码时也是肉眼可见的慢。

不过，我将继续深入研究机器学习算法，尤其是如何更高效地进行模型调优和如何结合不同模型的优势来提升整体性能。我相信，通过不断的学习和实践，我能在机器学习领域取得更大的进步，为解决实际问题提供更好的方案。