**《机器学习基础》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **年级、专业、班级** | | **2022级计算机科学与技术(卓越)01班** | | | **姓名** |  |
| **实验题目** | **对数几率回归算法实践** | | | | | |
| **实验时间** | **2024/04/7** | | **实验地点** | **DS3401** | | |
| **实验成绩** |  | | **实验性质** | **□验证性 ☑设计性 □综合性** | | |
| 教师评价：  **□**算法/实验过程正确； **□**源程序/实验内容提交 **□**程序结构/实验步骤合理；  **□**实验结果正确； **□**语法、语义正确； **□**报告规范；  其他：  评价教师签名： | | | | | | |
| 一、实验目的  掌握线性模型、对率回归算法原理。 | | | | | | |
| 二、实验项目内容  1. 理解对率回归算法原理。 2. 编程实现对数几率回归算法。 3. 将算法应用于西瓜数据集、鸢尾花数据集分类问题。 | | | | | | |
| 1. 实验过程或算法（源程序）   对数几率回归模型  本实验使用的 Logistics 回归模型虽然名为回归，但实际上是一种二分 类模型，由条件概率分布 P(y|x)表示。其中随机变量 x 为样本属性的向量，向量的每个元素描述一个样本的对应属性，随机变量 y 取值为 0 或 1，描述其分类标签。P(y=0|x)表示该样本为负类的概率，P(y=1|x)表示正类的概率。  对于每一个输入的样本属性向量 x，对其做一个高维到一维的映射 x->z：  其中w是转换矩阵，参数值需要训练得到；z是预测值，是一个实数。  对于二分类问题输出标签为y={0,1},因此需要将z映射成成 0/1 值。 最理想的 z->y 映射为“单位跃阶函数”，但考虑到其不具有连续、光滑、 可微等优良数学特性，采用对数几率函数替代：  该函数在 z=0 处变化很快，z>0 和 z<0时趋近于0/1，基本满足二分类任务的要求，并且具有良好的数学特性。  于是变换可得：  令P(y=1|x)=y，视为正类的概率；P(y=1|x)=1-y，视为负类的概率。其中y/(1-y) 为事件发生与不发生的概率的比值，称为几率，因此该模型称为对数几率。 解微分方程可得：  综上所述，输出 y=1 的对数几率是关于输入 x 的线性函数表示，即对数几率回归模型。  西瓜数据集代码：   1. *# 读取本地数据* 2. import csv 3. import numpy as np 4. *# 读取西瓜数据集中的数据并进行预处理* 5. def loadDataset(filename): 6. dataset = [] 7. labelset = [] 8. with open(filename, 'r', encoding='gbk') as csvfile: 9. csv\_reader = csv.reader(csvfile) 10. header = next(csv\_reader) 11. for row in csv\_reader: 12. if row[3] == '是': 13. labelset.append(1) 14. elif row[3] == '否': 15. labelset.append(0) 16. row[3] = 1 17. dataset.append(row) 18. data = [[float(x) for x in row] for row in dataset] 19. return dataset, labelset 20. *# 定义sigmoid函数* 21. def sigmoid(z): 22. return 1.0 / (1 + np.exp(-z)) 23. def test(dataset, labelset, w): 24. data = np.mat(dataset).astype(float) 25. y = sigmoid(np.dot(data, w)) 26. b, c = np.shape(y)  *# 功能是查看矩阵或者数组的维数。* 27. rightcount = 0 28. for i in range(b): 29. flag = -1 30. if y[i, 0] > 0.5: 31. flag = 1 32. elif y[i, 0] < 0.5: 33. flag = 0 34. if labelset[i] == flag: 35. rightcount += 1 36. rightrate = rightcount / len(dataset) 37. return rightrate 38. *# 迭代求w* 39. def training(dataset, labelset): 40. *# np.dot(a,b) a和b矩阵点乘* 41. *# np.transpose()  转置* 42. *# np.ones((m,n))  创建一个m行n列的多维数组* 43. data = np.mat(dataset).astype(float) 44. label = np.mat(labelset).transpose() 45. w = np.ones((len(dataset[0]), 1))  *# 初始化权重* 46. *# 步长* 47. n = 0.0001 48. *# 每次迭代计算一次正确率（在测试集上的正确率）* 49. *# 达到0.90的正确率，停止迭代* 50. rightrate = 0.0 51. while rightrate < 0.90: 52. c = sigmoid(np.dot(data, w)) 53. b = c - label *# 损失矩阵* 54. change = np.dot(np.transpose(data), b) *# 得到梯度* 55. w = w - change \* n 56. *# 预测，更新准确率* 57. rightrate = test(dataset, labelset, w) 58. return w 59. dataset = [] 60. labelset = [] 61. filename = '西瓜数据集3.0a.csv' 62. dataset, labelset = loadDataset(filename) 63. w = training(dataset, labelset) 64. print("若使得准确率大于90%，则此时的w为：\n", w) 65. print("正确率：%f" % (test(dataset, labelset, w) \* 100) + "%")   鸢尾花数据集代码：   1. import numpy as np 2. import pandas as pd 3. from sklearn.model\_selection import train\_test\_split 4. *# 读入数据集* 5. path = r'E:\python\Iris\iris-data.csv' 6. iris = pd.read\_csv(path, encoding='GBK') 7. iris['Species'] = iris['Species'].map({'setosa': 0, 'versicolor': 1, 'virginica': 2}) 8. label = np.array(iris['Species']) 9. data = np.array(iris[['SepalLength', 'SepalWidth', 'PetalLength', 'PetalWidth']]) 10. *# 引入偏置量* 11. b = np.ones((len(iris), 1)) 12. data = np.c\_[b, data] 13. *# 划分数据集为训练集和测试集* 14. X\_train, X\_test, y\_train, y\_test = train\_test\_split(data, label, test\_size=0.2, random\_state=100) 15. *# 定义函数* 16. def sigmoid(z): 17. return 1.0 / (1 + np.exp(-z)) 18. *# 优化参数* 19. w = np.ones((5, 3)) 20. accuracy\_train = [] 21. accuracy\_test = [] 22. for i in range(3):  *# 对每个类别进行训练* 23. yi\_train = np.where(y\_train == i, 1, 0)  *# 重新打标签* 24. yi\_test = np.where(y\_test == i, 1, 0)  *# 测试集的标签* 25. n = 0.01  *# 学习率* 26. iterations = 28  *# 迭代次数* 27. for j in range(iterations): 28. y = sigmoid(X\_train.dot(w[:, i])) 29. m = y - yi\_train 30. w[:, i] = w[:, i] - X\_train.transpose().dot(m) \* n / len(X\_train) 31. *# 在训练集上进行预测和计算准确率* 32. y\_pred\_train = sigmoid(X\_train.dot(w[:, i])) 33. y\_pred\_train[y\_pred\_train >= 0.5] = 1 34. y\_pred\_train[y\_pred\_train < 0.5] = 0 35. accuracy\_train.append(np.mean(y\_pred\_train == yi\_train)) 36. *# 在测试集上进行预测和计算准确率* 37. y\_pred\_test = sigmoid(X\_test.dot(w[:, i])) 38. y\_pred\_test[y\_pred\_test >= 0.5] = 1 39. y\_pred\_test[y\_pred\_test < 0.5] = 0 40. accuracy\_test.append(np.mean(y\_pred\_test == yi\_test)) 41. print(w) 42. print(f'训练集准确率为: {np.mean(accuracy\_train)}') 43. print(f'测试集准确率为: {np.mean(accuracy\_test)}') | | | | | | |
| 四、实验结果及分析  西瓜集结果：   1. D:\Python\python.exe "E:\python\pairwise regression\main.py" 2. 若使得准确率大于90%，则此时的w为： 3. [[-0.13238371] 4. [ 0.95912142] 5. [ 0.99030171] 6. [ 0.91692607]] 7. 正确率：94.117647% 8. Process finished with exit code 0   **数据加载和预处理：**从本地文件中读取数据集，将数据预处理成适合模型输入的格式，同时将标签转换为数字表示。  **Sigmoid 函数：**实现了 Sigmoid 函数，用于将预测结果映射到 (0, 1) 之间，作为概率值。  **测试函数：**用于在给定权重下测试模型在测试集上的准确率，通过计算预测结果与真实标签的对比得出准确率。  **训练函数：**采用梯度下降法更新模型权重，以使模型在测试集上的准确率达到一定要求（这里是 90%）。  **主程序：**加载数据集，调用训练函数训练模型，得到最终的权重，并输出最终的权重和模型在测试集上的准确率。  总体来说为了实现西瓜集的二分类，只需要将得到的分数通过Sigmoid函数转换为预测值，接着通过梯度下降算法迭代即可。  鸢尾花集结果：   1. D:\Python\python.exe E:\python\Iris\main.py 2. [[ 0.74348326  0.74348326  0.74348326] 3. [-0.48894363 -0.48894363 -0.48894363] 4. [ 0.21959573  0.21959573  0.21959573] 5. [ 0.03206665  0.03206665  0.03206665] 6. [ 0.69039866  0.69039866  0.69039866]] 7. 训练集准确率为: 0.9333333333333332 8. 测试集准确率为: 0.9666666666666667 9. Process finished with exit code 0   **数据准备：**首先，通过 Pandas 库读取了存储在 CSV 文件中的鸢尾花数据集，并将其转换为 NumPy 数组。然后，将类别标签转换为数字表示，并为数据集添加了偏置项（常数项），以便在模型中使用。  **数据集划分：**使用 train\_test\_split 函数将数据集划分为训练集和测试集，其中测试集占总数据的 20%。  **定义 Sigmoid 函数：**定义了 Sigmoid 函数，用于将预测结果映射到 (0, 1) 区间。  **参数优化：**针对每个类别，使用梯度下降法优化模型参数。迭代更新权重，直到达到指定的迭代次数。在每次迭代中，计算预测值并更新权重。  **模型评估：**在训练集和测试集上分别计算模型的准确率。通过将预测值与真实标签进行比较，计算准确率，并将结果存储在列表中。  因为鸢尾花集涉及到多分类问题，因此使用np.where给其重新打标签，从而将多分类问题转化为多个二分类问题。  通过实验发现，模型超参数、数据集样本数、数据集样本分布都会影 响模型的训练效果和准确率：  ·模型超参数：本实验中的模型超参数主要是训练次数和学习率。由于本 实验中学习率设置的较小，增加了训练时间，但对准确性影响不大。因此 此处主要分析训练次数对模型准确性：当训练次数较少时，模型训练参数 w 未收敛，因此准确率较低；随着训练次数增多，w 逐渐收敛，准确率不 断提高；当训练次数过多，一旦超过测试集拐点，可能会出现过拟合现象。  ·随机初始化矩阵：由于模型在初始化 w 时采用随机数，因此在同一超参 数下模型的训练准确率会上下波动。经实验发现，同一超参数下模型随机 初始化参数矩阵可能使准确率上下波动 10%。  ·数据集：对比一下同一模型在两个数据集上的训练效果：由于西瓜数据 集样本数量太少，分割训练集与测试集后几乎无法完成训练和测试，因此 将数据集同时作为训练集与测试集。但即便用训练集测试数据，因为数据太少的缘故，即使训练 10000000 次，准确率较低也难以超过 90%。反观 鸢尾花数据集，样本数较多，将其按 7:3 划分训练集和测试集，迭代 5000 次即可得到 100%的准确率。  值得一提的是，由于西瓜数据集只有 17 个样本，不仅训练效果不好， 在随机初始矩阵的影响下，训练结果也相当不稳定。即使较多迭代次数下 同一模型也可能得到相差较大的准确性，鸢尾花数据集就基本没有该不稳 定性。 其实对数几率回归及模型还有一定的改进空间，如加入正则化项防止 过拟合、softmax 推广 logistic 模型等。 | | | | | | |