人工智能基础

编程作业 2

http://staff.ustc.edu.cn/~linlixu/ai2018spring/

完成截止时间：2018/7/2

提交至：ustc_ai2018@163.com

助教： 赵若宇 [zry1997@mail.ustc.edu.cn]

梁润秋 [815556875@qq.com]

盛鑫 [xins@mail.ustc.edu.cn]

申书恒 [vaip@mail.ustc.edu.cn]

实验说明

目的

本次实验考虑机器学习中传统的监督学习问题与非监督学习，基于两个应用：垃圾邮件分类和 PCA 人脸识别，并结合课上介绍的相应学习算法，在数据集上分别进行实验，以加强对相关算法原理及应用的理解。

提交

1. 实验提交邮箱：ustc_ai2018@163.com，主题：学号_姓名_实验二

2. 附件格式为“学号_姓名.(rar|zip)”，要包括实验报告和实验代码。两个实验的文件分别放在 part1 和 part2 文件夹中。实验报告可以共用一份，也可以每个实验用单独的实验看 v 报告。

3. 实验使用 Python 语言

Part 1. 垃圾邮件分类(75%)

数据集介绍：SpamAssassin 语料库，是一个垃圾邮件语料库。其中的标签分别为 spam：垃圾邮件; easy ham：易识别的正常邮件。我们要通过监督学习训练一个**二分类分类器**，尽可能的实现垃圾邮件过滤功能。

提示思路：

1. 原始邮件预处理：去除头部，保留邮件正文。

2. 构建特征词类别知识库，通过从邮件正文中抽取特证词来构建邮件分类器的特征集。

语料库构造：量化特证词频率——构造一个词项-文档矩阵（TDM）N*M 的矩阵，[i, j]表示词项 i 在文档 j 中出现的次数。(这一步可以调用 python 里任何你熟悉的包)

3. 构建分类器。

训练与测试

在监督学习中，训练数据带有标号，在训练的过程中需要从训练数据 `traindata` 和其对应的标号 `trainlabel` 中学习相应的分类模型。

在测试过程中，用学习到的模型对测试集中的数据 `testdata` 作预测，并将预测结果与测试数据的真实标签 `testlabel` 进行比较，从而度量分类模型的性能。

为了有效评价我们训练出的模型的好坏，我们使用两个评价指标：`SP`（垃圾邮件识别的准确率，`Precision`），`SR`（垃圾邮件识别的查全率，`Recall`）：

$$\mathrm{SP} = \frac{n_{pam \to pam}}{n_{pam \to pam} + n_{normal \to pam}}$$

$$\mathrm{SR} = \frac{n_{pam \to pam}}{N_{pam}}$$

这里$n_{pam \to pam}$是正确识别的垃圾邮件数；$n_{normal \to pam}$是正常邮件被误识别为垃圾邮件数；N_{pam}是垃圾邮件总数。

SP，SR 反映了模型质量的两个方面。为了综合考虑模型的性能，我们引入一个新的评价指标 **F 值**，它综合考虑了准确率和查全率两方面的指标，F 值越大，模型越优，其数学公式如下：

$$\mathrm{F} = \frac{SP \times SR \times 2}{SP + SR}$$

实验要求：

1．实现一个朴素贝叶斯分类器(15%)

提交一个 Python 函数 nBayesClassifier(traindata, trainlabel, testdata, testlabel, threshold)，其中 threshold 为用于判断类别的后验概率的阈值。要求函数返回对测试数据的预测 ypred，以及通过与 ground label(真实标签)比较计算得到的性能指标 SP,SR 和 F。ypred 与 SP,SR,F 以 tuple 形式返回。我们要求以表格的形式给出你的结果(其余两个分类器要求类似)并对分类器性能做出分析。

2．实现一个最小二乘分类器(引入规范化项后)(10%)

1)．对引入了 L2 规范化项之后的最小二乘分类问题进行推导。即求解以下优化问题：

$$min_w(X\boldsymbol{w} - y)^2 + \lambda\|\boldsymbol{w}\|^2$$

2)．基于 1 中的结果，实现并提交一个 Python 函数 lsClassifier(traindata, trainlabel, testdata, testlabel, lambda)

3．实现一个支持向量机分类器（15%）

提交一个 Python 函数 softsvm(traindata, trainlabel, testdata, testlabel, sigma, C)，其中 C 为 soft margin SVM 的控制参数，sigma 为控制核函数的参数，当 sigma=0 时，使用线性核函数$\mathrm{K}(x_i, x_j) = {x_i}^T x_j$,其他情况则使用 RBF 核函数$\mathrm{K}(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{sigma^2}}$

4．在不同数据集上使用交叉验证选择各个算法的参数(15%)

实现交叉验证（代码需要提交），在各个数据集上：

- 使用 5-fold 交叉验证为每个算法挑选适当的参数(Naïve Bayes 中的 threshold，最小二乘法中的 Lambda，SVM 中的 sigma 和 C)；
- 对每一个算法：
  - 返回一个矩阵，表示每一个参数（参数组合）在每一个 fold 上的 F 值（若有 10 个参数，则返回 10x5 的矩阵）；
  - 挑选在 5 个 fold 中平均 F 值最高的参数（参数组合）

在实验报告中需要记录交叉验证的结果，即对于每个参数(参数组合)在 5 个 fold 上的平均 F 值。

5．实验报告(20%)

总结以上的实验结果，并对实验结果进行分析。如果你训练出的分类器效果不好，

想办法改进你的分类器，例如改善特征选取方法。比较改进前后模型的变化并对你的改进做出分析，我们会根据你的分析适当加分。

Part 2. 基于 PCA 的人脸识别技术(25%)

在这部分实验中，我们将完成人脸识别。首先利用训练集训练出一个人脸模型，然后将测试样本与训练集进行匹配，找到与之对应的训练集人脸类别。最直接的方式是利用欧式距离计算测试集的每一幅图像与训练集的每一幅图像的距离，然后选择距离最近的图像作为识别的结果。这种直接计算距离的方式直观，但是有明显的缺陷，包括计算量太大，高维灾难等问题。

解决上述问题的一个途径是对图像进行降维，通过只保留某些关键像素可以使识别速度大大提升。降维的一个方法即是 PCA（主成分分析）。我们在此实验中将使用 PCA 对图像进行了降维，以提高人脸的匹配效率及效果。

数据集及数据处理说明：

1. 我们提供 40 个人的人脸图片，每个人 10 张，共 400 张，相同的人脸图片在同一文件夹下，图像尺寸是 92×112，图像背景为黑色。
2. 将每个文件夹随机取出两张头像（40个文件夹共80张）作为测试集，其余头像作为训练集，将测试集中的每一幅降维图像与降维的训练集进行匹配，然后将其分类到距离最小的训练集头像中，如果两个头像表示一个人，表示识别成功，否则表示识别失败。并计算出识别率。

实验要求：

1.实现一个PCA降维算法（15%）

实现并提交一个Python函数PCA（traindata, testdata, threshold）

其中threshold表示特征值的累计贡献率。即选择前m个特征向量，使得

$$\frac{\text{Sum(first m} - 1 \text{ eigenvalues)}}{\text{Sum(all eigenvalues)}} < threshold \leq \frac{\text{Sum(first m eigenvalues)}}{\text{Sum(all eigenvalues)}}$$

2.实验验证 PCA 算法效果及实验报告（10%）

a）检验随着 threshold 不同取值，PCA 选择的降维维度和识别准确率会有什么样的变化。

b）仔细观察，并尝试提出算法的改进方案。

c）在实验报告中总结以上的实验结果。

d）提交将训练集降维后的图像矩阵。

备注：

1．矢量化编程是提高算法速度的一种有效方法，其思想就是尽量使用高度优化的数值运算操作来实习学习算法。在 Python 中，我们通常使用 NumPy 来进行向量与矩阵的运算。例如，假设为向量，需要计算，在 Python 中可以用以下方式实现：

```
z = 0
for i in range(0,n)
   z = z + x[i] * y[i];
end
```

或者可以更简单的写为：

```
import numpy as np
z = np.dot(x, y);
```

很显然，第二段程序代码不仅简单，而且运行速度更快。

特别是对于核函数的计算，希望能够尽量使用矢量化编程的思想来减小计算复杂度，我们将根据算法的优化进行相应加分。

2.Naive Bayes 算法中的 threshold 的取值可以从[0.5 0.6 0.7 0.75 0.8 0.85 0.9]中取值；最小二乘分类器中的 lambda 可以从[1e-4 0.01 0.1 0.5 1

5 10 100 1000 5000 10000]中取值；SVM 中的参数有高斯核参数 sigma 以及 C，其中 sigma 的取值范围由数据决定：假设数据集为$\{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_n}, y_n)\}$，令 $\mathrm{d} = \frac{\sum_{i,j}(\mathbf{x_i} - \mathbf{x_j})^2}{n^2}$，则 sigma 从[0.01d 0.1d d 10d 100d]中取值，C 可以取[1 10 100 1000]。

3.对于实验 2 的向量之间距离度量，可以使用欧氏距离或者曼哈顿距离等，自由选取。

4. PCA 算法要求自己实现。要求代码对任意训练集都可以训练出好的效果，主代码放在脚本 PCA.py 中。代码要简洁工整，尽量使用向量化编程（见备注 1），必要的地方要有注释。