题目　Ensemble Learning.

姓名 骆克云 学号 MG1633052 邮箱 streamer.ky@foxmail.com 联系方式18115128082

(南京大学 计算机科学与技术系, 南京　210093)

1　实现细节

1.1　读取文件，返回数据和标签（projectutil.py）

使用 Python 读取文本文件：读取第一行数据作为特征类型单独保存，将其余每一行数据按逗号分隔，最后一个作为标签，前面的作为数据

```
def get_ensemble_data(name="breast-cancer"):
    """
    获取数据集
    :param name:数据名称: dataset1-a9a/covtype
    :param type:数据类型: train/test
    :return:数据，标签
    """
    file = project_dir + os.sep + "data/EnsembleLearning/" + name + "-assignment5.txt"

    feature_types = None
    data = []
    label = []
    first_row = True
    with open(file, "r", encoding="utf-8") as f:
        for line in f:
            line = line.split(",")
            if first_row:
                feature_types = line
                first_row = False
                continue
            data.append(line[:-1])
            label.append(line[-1].strip("\n"))

    return data, label, feature_types
```

1.2　K折交叉验证(assignment5/kfolddataset.py)

将数据打乱随机均分成 K 组，每次选取其中的一份作为测试集，其余的作为训练集，这样相当于得到了 K 个数据集，然后对这 K 个数据集进行分类等计算。

```python
def k_fold_train_test_split(k, data):
    """K折交叉验证：训练集/测试集数据划分"""
    m = np.arange(len(data))
    np.random.shuffle(m)
    train_test_slices = np.array_split(m, k)
    slices_array = np.zeros(0)

    train_test_data = [[] for i in [0, 1]]
    for i in range(k):
        """测试集数据"""
        slices = train_test_slices.copy()
        for j in range(k):
            if j != i:
                slices_array = np.append(slices_array, slices[j])
        train_test_data[1].append(np.array(slices[i]).astype(int))
        train_test_data[0].append(slices_array.astype(int))
    return train_test_data
```

1.3 朴素贝叶斯算法实现（assignment5/naivebayes.py）

朴素贝叶斯算法使用概率生成模型，生成属于某一类别的概率。对于离散特征，使用条件概率和先验概率相乘：$Pr(y)\cdot\prod_i Pr(x_i|y)$，并使用拉普拉斯平滑，对于连续特征，假定其满足高斯分布，使用先验概率下的高斯密度函数作为概率值。

具体算法如下：

1）初始化(__init__， fit)：定义成员变量，并初始化；

2）高斯函数(def gaussian_func(self, feature, label, x))：计算指定特征，给定标签下的数据 X 的高斯密度函数值；

3）训练数据集(def train_data(self))：对于每一类标签，计算其先验概率，对于离散特征，计算其在该标签下每个特征取值的概率，计算时使用拉普拉斯平滑（分子加 1，分母加 2）。对于连续特征，计算其高斯密度函数，最后将它们相乘(贝叶斯假设)；

4）计算测试集属于某类的概率(def calculate_prob(self, data, label))：根据朴素贝叶斯公式，将测试集中的每一个特征相乘，再乘上先验概率，计算其属于某类的概率；

5）训练集预测(def train_predict(self)):对于训练集数据，计算每个数据属于某个类的概率，取概率值大的类作为该类，本步骤是 Adaboost 方法调用的；

6）测试集预测(def train_predict(self)):对每个测试集中的数据，计算每个数据属于某个类的概率，取概率值大的类作为该类；

7）测试集准确度计算(def score(self, filename, pred, alg="Naive Bayes"):)：比较预测结果和真实标签，计算准确度；

8）运行接口(def run(self, filename)):运行朴素贝叶斯算法。

1.4 AdaBoost提升算法实现（assignment5/adaboost.py）

Adaboost 利用弱分类器反复学习改变训练数据的权值分布得到一系列的弱分类器算法，然后将它们组合

起来提升为强学习算法。本次实验要求使用朴素贝叶斯作为弱分类器。

具体步骤如下：

1) 将标签标准化为-1 和 1，初始化训练数据 train_data 的权值分布：初始化权值向量 $D_1=(w_{1,1}, w_{1,2},...,w_{1,n})$, $w_{1,i} = 1/N$,;

2) 循环 m 次： m = 1,2,...,M:

2a) 利用朴素贝叶斯分类器分类得到训练集和测试集的预测值 F_m，计算训练集分类出现错误的数据对应的权值向量的和，即错误率 error,计算分类器系数$\alpha_m=0.5*\log((1 - error) / error)$;

2b) 更新训练数据集的权值分布 $D_{m+1} = (w_{m+1,1}, w_{m+1,2},...,w_{m+1,n})$, $w_{m+1,i} = w_{m,i} / Z_m *e^{-\alpha myiGm(xi)}$, i=1,2,...N, Z 是归一化因子：$Zm = sum(w_{m,i} *e^{-\alpha myiGm(xi)})$;

2c) 根据新的权值分布 D_{m+1} 对训练数据按照概率采样组成新的训练集。

3) 构建基本分类器的线性组合：$f(x) = sum(\alpha_m F_m)$, 对于 f(x)中的每一项，若值大于或等于 0 则为 1，否则为-1.

运行时，将数据进行 10 折交叉验证，对每一份数据运行上述算法，得到测试集的准确率，计算准确率的均值和标准差。

1.5 运行

对于每个文件（"breast-cancer", "german"），运行 Adaboost，得出结果。

```
1. def run():
2.     for file in ["breast-cancer", "german"]:
3.         adaboost = AdaBoost()
4.         adaboost.run(file)
```

2 结果

2.1 实验设置

数据来源：EnsembleLearning,包含如下文件：german-assignment5.txt ， breast-cancer-assignment5.txt

数据预处理：projectutil.py 中 get_ensemble_data(name="breast-cancer")可给定文件名返回数据及标签以及特征类型。

Adaboost 提升学习次数 M=5。

2.2 实验结果

1. Adaboost：K=10 折交叉验证

K	1	2	3	4	5	6	7	8	9	10	均值	标 准差
breast-canc	0.5714	0.6786	0.7857	0.9286	0.8571	0.8571	0.6429	0.7407	0.8148	0.7778	0.7655	0.1036

er												
germa n	0.69	0.79	0.70	0.71	0.75	0.82	0.79	0.77	0.74	0.82	0.758	0.045 3

2. 结果反思

朴素贝叶斯作为概率生成模型分类器，对 Adaboost 的提升效果并不好。

注意：

1. 最终提交的报告最好保存为 pdf 格式
2. 压缩格式为 zip 格式，请勿使用需要安装特定软件才能打开的压缩方式
3. 作业的文件夹目录请按照网页要求，代码、结果放在不同子文件夹中。作业网页上给出的数据不需要再次提交