华南理工大学

South China University of Technology

# 《机器学习》课程实验报告


学　　院　软件学院

专　　业　软件工程

组　　员　郑铭莉、陈佳欣、袁修竹

学　　号　201530613801

201530611180

201530613535

邮　　箱　727457905@qq.com

指导教师　吴庆耀


提交日期　2017 年 12 月 22 日

**1. 实验题目：基于 AdaBoost 算法的人脸检测**

**2. 实验时间：2017 年 12 月 16 日**

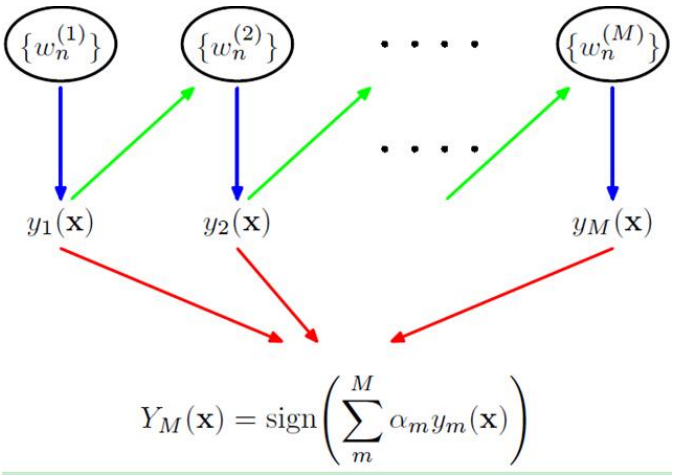
**3. 报告人：郑铭莉、陈佳欣、袁修竹**

**4. 实验目的:**

深入理解 Adaboost 的原理
熟悉人脸检测的基本方法
学会利用 Adaboost 解决人脸分类问题，将理论和实际工程接轨
体验机器学习的完整过程

**5. 简述 Adaboost 原理：**

AdaBoost 是 Boosting 方法中最优代表性的提升算法。该方法通过在每轮降低分对样例的权重，增加分错样例的权重，使得分类器在迭代过程中逐步改进，最终将所有分类器线性组合得到最终分类器，Boost 算法框架如下图所示：




**Algorithm 2:** Adaboost

**Input:** $D = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$, **where** $\mathbf{x}_i \in X, y_i \in \{-1, 1\}$
**Initialize:** Sample distribution $w_m$
**Base learner:** $\mathcal{L}$
$w_1(i) = \frac{1}{n}$
**for** m=1,2,...,M **do**
  $h_m(x) = \mathcal{L}(D, w_m)$
  $\epsilon_m = \sum_{i=1}^{n} w_m(i)\mathbb{I}(h_m(\mathbf{x}_i) \neq y_i)$
  **if** $\epsilon_m > 0.5$ **then**
    **break**
  **end**
  $\alpha_m = \frac{1}{2}\log\frac{1-\epsilon_m}{\epsilon_m}$
  $w_{m+1}(i) = \frac{w_m(i)}{z_m} e^{-\alpha_m y_i h_m(\mathbf{x}_i)}$, **where** $i = 1, 2, ..., n$ **and** $z_m = \sum_{i=1}^{n} w_m(i) e^{-\alpha_m y_i h_m(\mathbf{x}_i)}$
**end**
**Output:** $H(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m h_m(\mathbf{x})$

## 6. 数据集以及数据分析：

本实验提供 1000 张图片，其中 500 张是含有人脸的 RGB 图片，储存在./datasets/original/face 内；另外 500 张是不含有人脸的 RGB 图，储存在./datasets/original/nonface 内。

数据集包含在示例仓库内，请自行下载并将其切分为训练集，验证集。

## 7. 实验步骤:

1.读取数据集数据。读取图片，将全部图片转成大小为 24*24 的灰度图，数据集正负类样本的个数和比例不限，数据集标签形式不限。

2.处理数据集数据，提取 NPD 特征。使用 feature.py 中 NPDFeature 类的方法提取特征。（提示：因为预处理数据集的时间比较长，可以用 pickle 库中的 dump()函数将预处理后的特征数据保存到缓存中，之后可以使用 load()函数读取特征数据）

3.将数据集切分为训练集和验证集，本次实验不切分测试集。

4.根据 ensemble.py 中的预留的接口编写 AdaboostClassifier 所有函数。以下为 AdaboostClassifier 类中的 fit()方法的思路：

4.1 初始化训练集的权值 $\omega$ ,每一个训练样本被赋予相同的权值。

4.2 训练一个基分类器，基分类器可以使用 sklearn.tree 库中 DecisionTreeClassifier(注意训练的时候需要将权重 $\omega$ 作为参数传入)。

4.3 计算基分类器在训练集上的分类误差率 $\epsilon$ 。

4.4 根据分类误差率 $\epsilon$，计算参数 $\alpha$ 。

4.5 更新训练集的权值 $\omega$ 。

4.6 重复以上 4.2-4.6 的步骤进行迭代，迭代次数为基分类器的个数。

5.用 AdaboostClassifier 中的方法在验证集上进行预测并计算精确率,并用 sklearn.metrics 库的 classification_report()函数将预测结果写入 report.txt 中。

6.整理实验结果并完成实验报告。

## 8. 代码内容:

训练过程：

```
for i in range(0, maxIteration):
    print("Iteration",i)
    clf_tree.fit(img_features_train, img_label_train, sample_weight=weights)
    hypothesis.append (clf_tree.predict(img_features_train) )

    miss = [int(x) for x in ( hypothesis[i] != img_label_train )]
    miss2 = [x if x==1 else -1 for x in miss]
    err_m = np.dot(weights,miss)
    if(err_m > 0.5):
        break
    print('error: ',err_m)
    alpha_m.append( 0.5 * np.log( (1 - err_m) / float(err_m)) )
```

```
            weights = np.multiply(weights, np.exp([float(x) * alpha_m[i] for x in
miss2]))
            weights_sum = weights.sum()
            weights = weights / weights_sum
            prediction = prediction + alpha_m[i] * hypothesis[i]
            print('train_accuracy:
',get_accuracy(np.sign(prediction),img_label_train))

            validation_pred = clf_tree.predict(img_features_validation)
            validation_prediction = alpha_m[i] * validation_pred
            target_names = ['class 0', 'class 1']
            acc              =              get_accuracy(np.sign(validation_prediction),
img_label_validation)
            accuracy.append(acc)
            print(classification_report(img_label_validation,
np.sign(validation_prediction), target_names=target_names))
```

## 9. 实验结果和曲线图:

### 超参数选择（弱分类器个数、决策树深度等）：

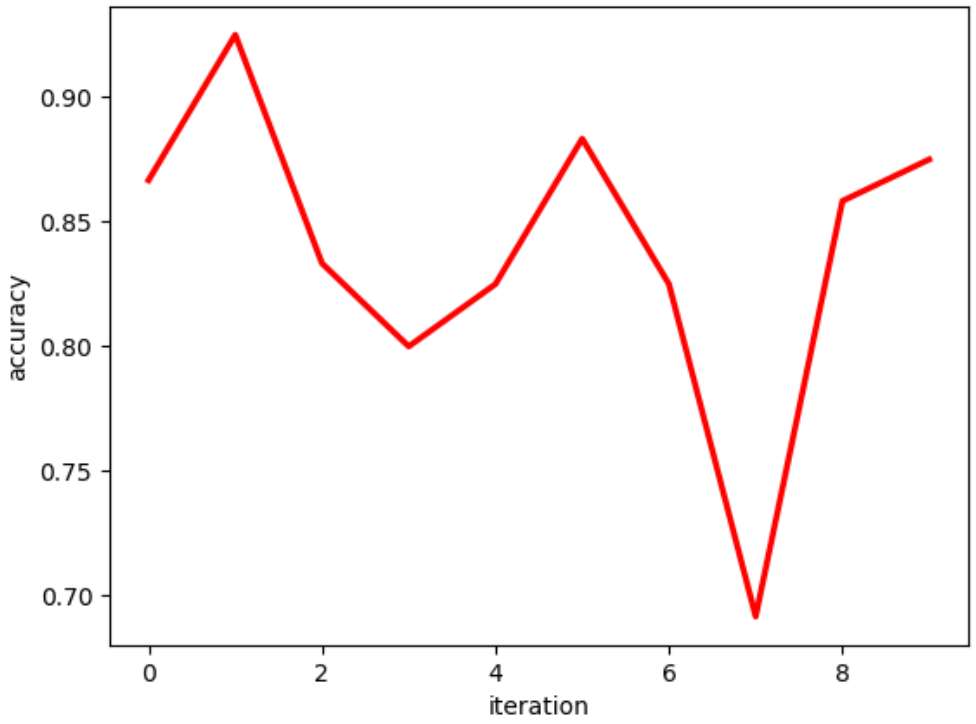
分类器数：2
深度：3

### 预测结果（最佳结果）：

error:  0.00724637681159
train_accuracy:  0.985714285714

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| class 0 | 0.90 | 0.95 | 0.93 | 60 |
| class 1 | 0.95 | 0.90 | 0.92 | 60 |
| avg / total | 0.93 | 0.93 | 0.92 | 120 |

精度曲线图：




## 10.实验结果分析：

训练集随着分类器数量增加，准确率逐渐提高，甚至达到 1。但在验证集上的精确率在主要集中在 0.7 和 0.8 之间，当为两个分类器时达到精确率达到 0.93。此外，深度达到 4 时出现计算溢出，因为 error 开始变为 0。

## 11.实验总结：

对 adaboost 算法有了深刻的认识，它的分类效率很高，效果很好，但是对于决策树深度的认识还有疑惑，当深度在 4 之后都会出现计算溢出，原因还在查找。