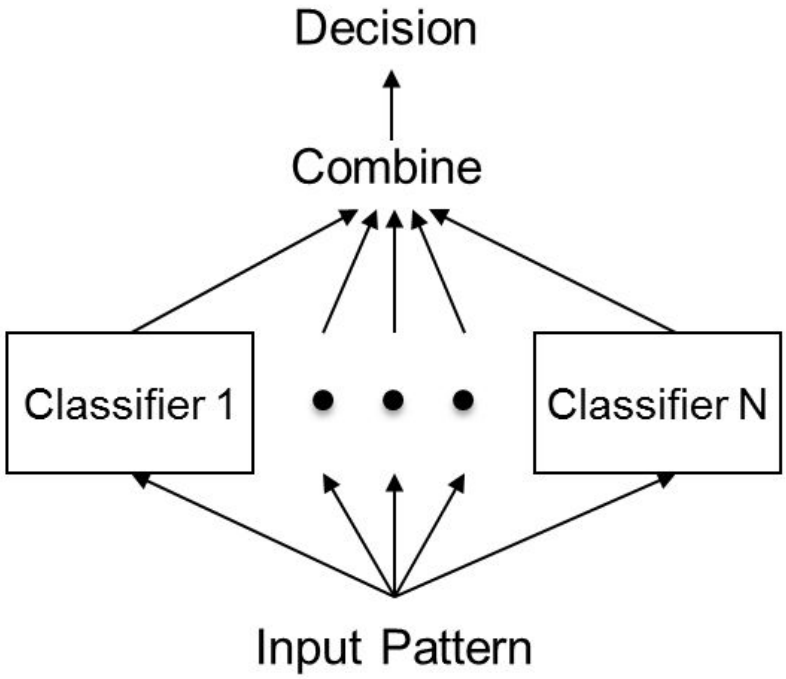
2016~2017春季学期机器学习概论

集成学习实验

实验报告



计43

唐玉涵

2014011328

2016~2017春季学期机器学习概论

集成学习实验

实验报告

1、任务说明

在本实验中，我们需要完成集成学习算法的实现，并且在一组真实的数据集上测试它的性能。具体包括：Bagging + DTree, Bagging + SVM, AdaBoost.M1 + DTree, AdaBoost.M1 + SVM 这四个基础的任务，并讨论其性能优劣。

2、实验设计

我采用外部库调用比较方便的python编程语言进行本次实验。所有源代码均在 exp2.py 文件之中，主要使用的外部库为 scikit-learn ，需要提前安装 python 包：sklearn，numpy 和 scipy。

实验基础部分完成了 Bagging 和 AdaBoost 两种集成学习算法，并分别在决策树 Dtree 和 SVM 上进行了试验，对比两种集成学习方法的性能提升。拓展部分研究了朴素贝叶斯在两种集成学习方法下的表现，并讨论了特征选择对分类结果的影响。

具体来说，分为数据读入，数据处理，进行训练，评估性能四个部分。拓展部分放在之后单独进行说明。

首先按行读入csv文件，储存原始数据，然后把数据分为特征和分类，即X和y，装入不同的list之中。之后对X进行标准化处理，并按照实验要求进行训练集和测试集的划分。进行训练时，根据不同的task，分别对应 Bagging + DTree, Bagging + SVM, AdaBoost.M1 + DTree, AdaBoost.M1 + SVM 这四种算法，其中DTree和SVM均是采用sklearn包内的实现，即调用 svm.LinearSVC(dual=False) 和 tree.DecisionTreeClassifier() 函数，自己实现了Bagging和AdaBoost算法。

为了对分类的性能做出评价，我采用了准确率作为衡量指标，即

- $$Accuracy = \frac{\text{number of correctly classified records}}{\text{number test records}}$$

统计针对全体测试集数据的正确预测个数，除以测试集大小，即可得到该方法的准确率。

3、实验结果

上述方法四种方法实现的集成学习分类器准确率见下表：

	DTree	SVM
Bagging	88.174%	73.612%
AdaBoost	84.553%	73.612%

可见整体而言决策树比SVM在本试验任务上的表现要好，准确率明显高出不少，而对于Bagging的效果优于AdaBoost，且差异明显；对于SVM两种集成算法的影响不大，准确率几乎一致。

对其原因的分析和讨论以及拓展内容详见下一部分。

4、实验分析与拓展

1. 实验基础部分结果分析

决策树与SVM相比，通过了集成学习，效果明显要好。这与理论的预测分析也比较吻合，因为决策树的模型是不稳定模型，受数据的影响较大，而集成学习正是对这种不稳定模型有更好的提升作用。

Bagging与Adaboost相比，两者都是集成学习的算法，均是针对单钟学习器进行整合集成，可提高尤其是不稳定学习器的学习效果。但Bagging算法在训练时采用Bootstap拔靴法采样，故每次的训练集其实都不同，Boosting算法每次的训练集样本是一样的，但是赋予了每一个样本不同的权值。就决策树而言，Bagging的提升效果更

好，可能的原因是Bootstrap采样提升了其算法的稳定性，若是对样本进行过采样，则可能两种集成学习方法的效果不会差别很大。

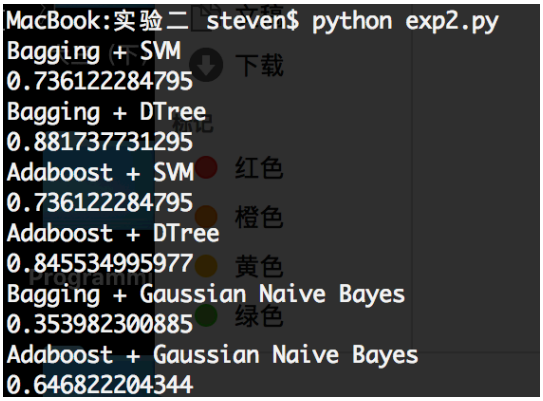
在实验中，表现最好的分类器是Bagging + DTree，准确率达到了88.174%，这也与理论预测的最优结果比较相符，Bagging算法对决策树的缺陷进行了很好地弥补，以致学习效果大大改善。

2. 实验拓展

(1) 其他分类器

为了验证集成算法对于不稳定模型的效果要与稳定模型，我选用了朴素贝叶斯方法进行对比实验，即调用GaussianNB() 函数进行朴素贝叶斯分类，可能与未进行调参有关，Bagging + Gaussian Naive Bayes 的效果非常差，Adaboost + Gaussian Naive Bayes 的结果要好很多。

实验结果见下：



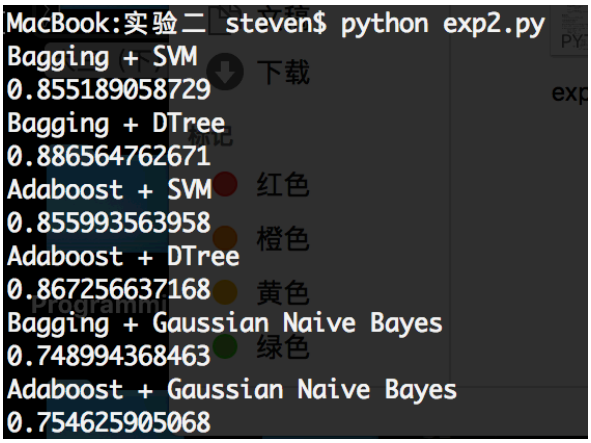
即Bagging + Gaussian Naive Bayes准确率只有35.398%，而Adaboost + Gaussian Naive Bayes 可以达到64.682%，这样验证了集成学习对于朴素贝叶斯这样的稳定模型而言，提升效果并不好。

(2) 特征选择对分类结果的影响

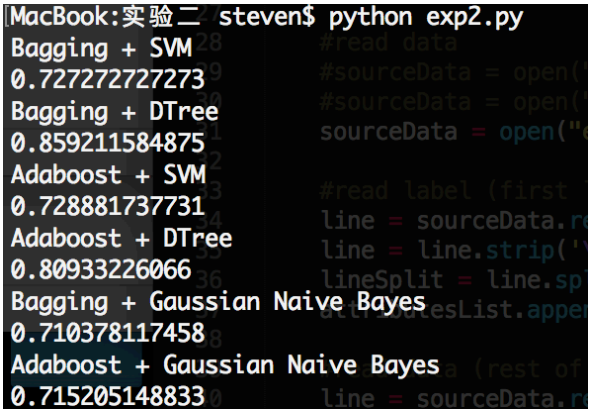
实验数据集的样本中，有96个content features和138个transformed link features，在本次实验中，我也尝试了选择只用content features或者transformed link features，效果有所提升。

具体结果如下：

只使用transformed link features的结果：



只使用content features的结果：



分析：

对比之前的使用全部特征的测试结果发现，对于SVM和DTree，均有性能："只使用transformed link features"优于"使用全部特征"优于"只使用content features"，而对于朴素贝叶斯，则是"只使用transformed link features"优于"只使用content features"优于"使用全部特征"，且Bagging + Gaussian Naive Bayes 的效果从35.398%提升到了74.899%，提升非常明显。

分析以上结果，可知并非特征使用越多越好，有些特征的加入反而会影响分类器的性能，在本实验中transformed link features对于分类而言具有比较积极的意义，而content features则显得不那么重要，甚至有一些冗余，分析其实际使用背景也可知对于一封邮件，它的长短字数等内容特征可能对于其是否为垃圾邮件的帮助不是很大。

比较喜人的人通过选取部分特征的方法，朴素贝叶斯分类器的性能有了很大提升，这启示我们对于朴素贝叶斯方法，要尽量避免冗余特征的加入，因为它会大大影响到实验的结果。

综合以上所有结果，准确率最高的是只使用transformed link features的Bagging + DTree，准确率可达到88.656%。

5、实验小结

本次实验中，我第一次在真实数据集上实现了集成学习的机器学习方法，并且学会了调用外部开发包进行机器学习编程，大大简化了编程复杂度，也提升了效率。在对实验的各种结果进行分析的过程中，我也逐渐学会了分析机器学习分类器性能的方法。

实验基础部分结束后，我还对一些可能的拓展进行了一些小尝试，也取得了不错的结果，采用部分特征后整体分类器的性能和效率都有所提升。并且通过自己的真实实验验证了集成学习对于不稳定模型的提升更大这一理论结果，有助于自己加深对于机器学习理论方面的理解。

最后，这次实验令我收获很大，不仅自己动手实现了具有实际意义的集成学习分类算法，更让我意识到算法写好之后的优化和调整往往也是十分重要的，更重要的是去主动思考每一种方法对实验结果产生相应影响的原因。同时，十分感谢张敏老师和张帆助教对我的帮助。因为种种原因没能按时提交，为助教学长带来了很多的麻烦，在此真诚地说一声抱歉。