集成学习实验报告

计 24 2012011335 柯均洁

一、实验内容

Compare different ensemble learning algorithms with different base classifiers.

- Ensemble Learning algorithm
  1. Bagging
  2. AdaBoost.M1
- Base classifier
  1. Decision Tree
  2. SVM
  3. Naïve Bayes
  4. Perceptron

二、实验设计

- **数据处理**
  1. Feature Selection

     分别尝试有 link feature 和无 link feature 下的分类，比较结果的差异
  2. Split into Train/Test Dataset

     随机从数据集中选取 4/5 的数据作为训练集，剩下的作为测试集
  3. Overcome Unbalanced Dataset

     原数据集中 normal 与 spam 的数量比大概是 2.4，也就是说存在比较严重的数据不平衡的问题，需要进行一定的预处理
     - Stratified Sampling

       将数据集分为 train/test 的时候将 spam 和 normal 的数据分开进行 sample。保证在 train、test 中的 spam 和 normal 所占比例和原数据集中的比例一致

- Multiply spam samples

  划分好训练集和测试集后，随机选取训练集中的 spam 例子进行复制，使得训练集中的 spam 和 normal 类的比例为 1:1

4. Feature Normalization

   对原数据的 feature 进行标准化，并比较标准化前后正确率的差异

- **算法实现(ensemble.py)**

实现了 Bagging 和 AdaBoostM1 的集成学习算法，用 Python Scikit-Learn Library (scikit-learn.org)实现 base classifier 的算法

- Bagging
  a) bootstrapping(self, x, y)

     用 bootstrap 的方法对训练集进行放回抽样

  b) train(self, x, y, basic_classifier)

     对 x, y 进行 bootstrap 抽样，并用 basic_classifier 对抽样后的数据进行训练，将新训练出来的 basic_classifier 插入到分类器列表中

  c) predict(self, x)

     用分类器列表中的每个分类器对 x 进行分类，并投票决定 x 的分类

- AdaBoostM1
  a) __init__(self, sampleNum)

     初始化，将所有 sample 的权值 weight 设为 1/sampleNum。

  b) train(self, x, y, basic_classifier)

     用 basic_classifier 对加了权值 weight 后的 sample 进行训练，计算错误率 Et，计算 beta，更新权值，并保存当前分类器的 voting weight

  c) predict(self, x)

     用分类器列表中的每个分类器对 x 进行分类，根据 voting

weight 的大小来投票决定 x 的分类

$$c^*(x) = argmax_{c^m} \sum_{c_t(x)=c^m} \log(1/\beta_t)$$

- **结果评价**

  分别计算了每一轮 ensemble learning 分类 Accuracy, Precision, Recall, F1 Score。并绘制曲线

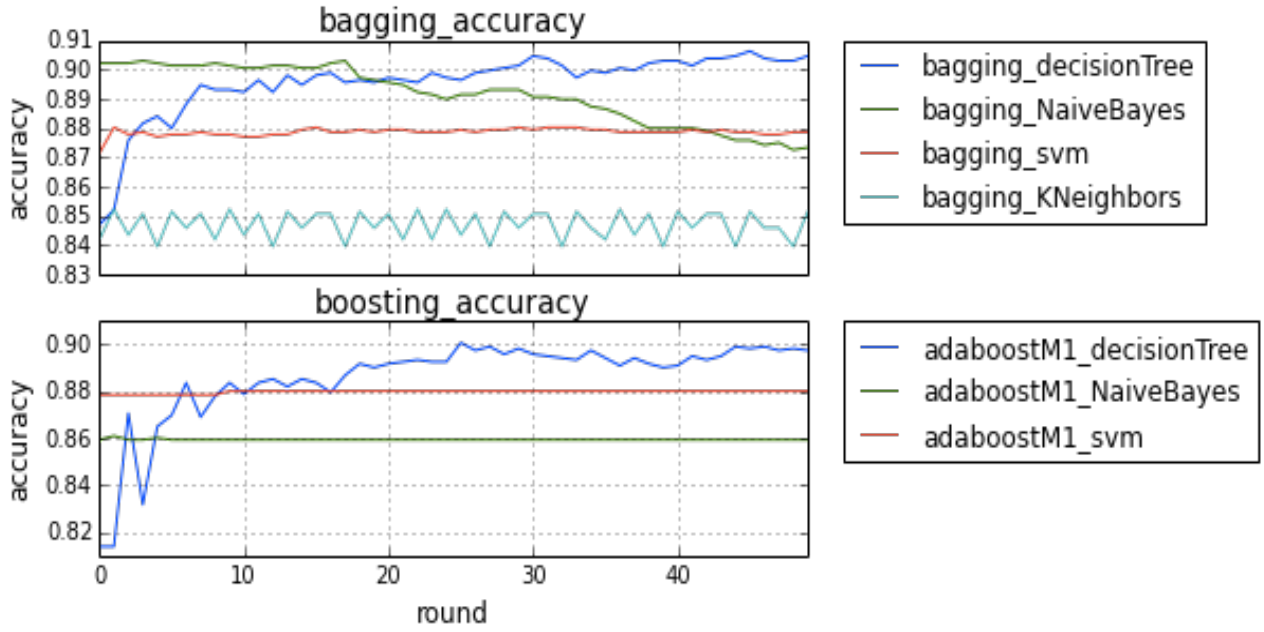
  - Accuracy = number of correctly classified / number of test case
  - Precision = True Positive/ (True Positive + False Positive)
  - Recall = True Positive / (True Positive + False Negative)
  - F1 score = 2 * Precision * Recall / (Precision + Recall)

三、实验结果

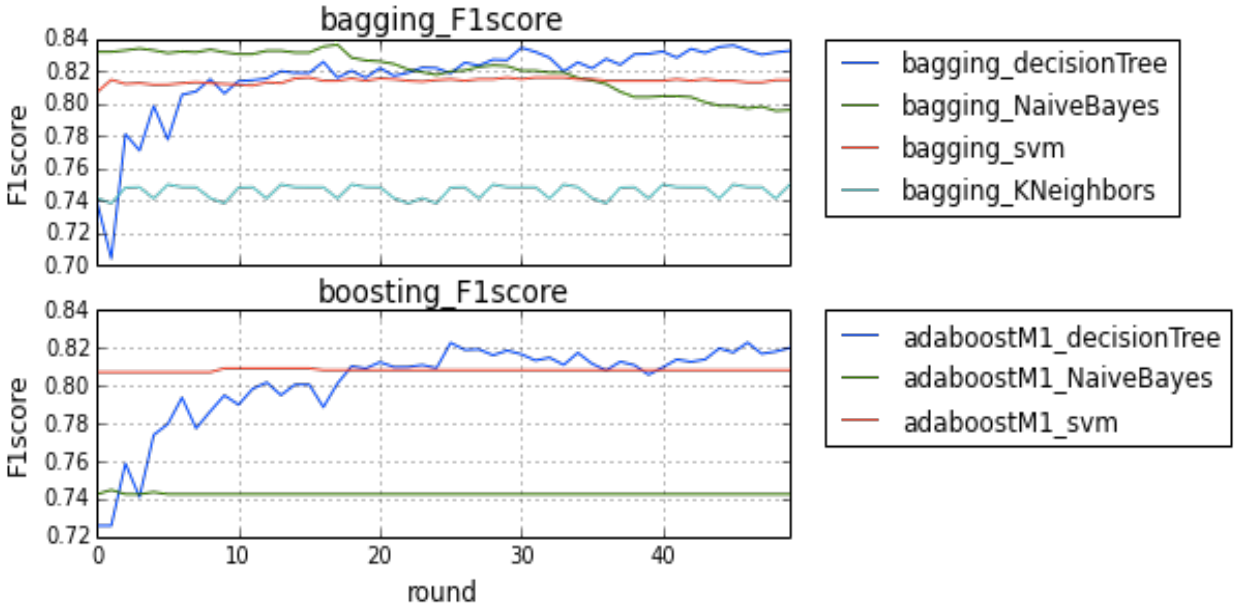
- 有 Link Feature 及 Feature Normalization
  - 50 轮 ensemble 后最终结果：

Method	Accuracy	Precision	Recall	F1 Score
bagging_decisionTree	90.4605	80.6723	85.9701	83.237
bagging_naiveBayes	87.3355	71.599	89.5522	79.5756
bagging_svm	87.8617	72.9075	92.2006	81.4268
Bagging_KNeightbors	85.1469	72.5389	77.5623	74.9665
adaboostM1_decisionTree	89.7561	77.3585	87.234	82.0
adaboostM1_naiveBayes	85.9388	73.1343	75.3846	74.2424
adaboostM1_svm	88.0192	73.1481	90.2857	80.8184

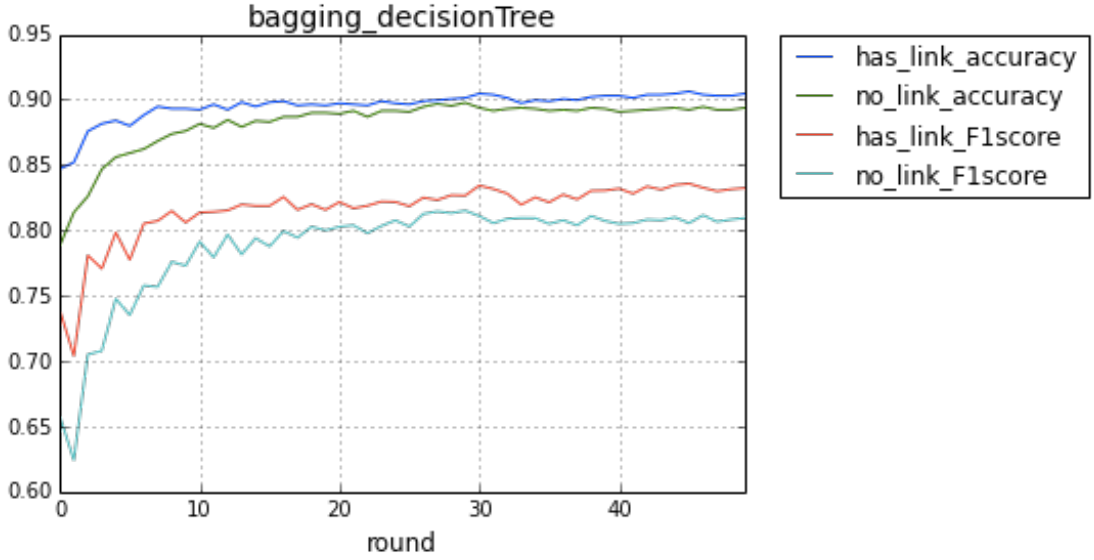
  - Accuracy 及 F1 score 指标随 ensemble 轮数的变化情况：
    - Accuracy

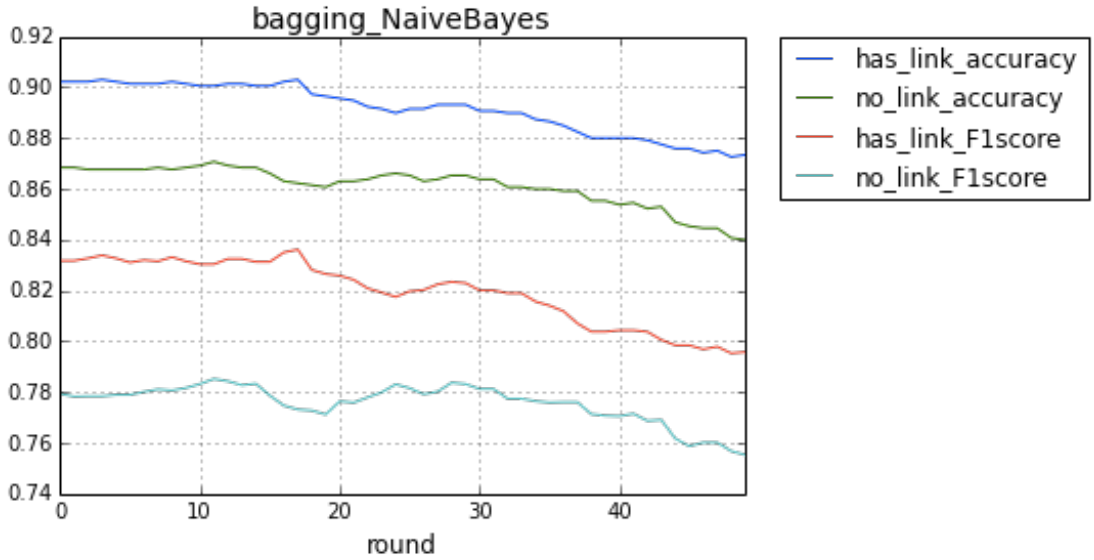


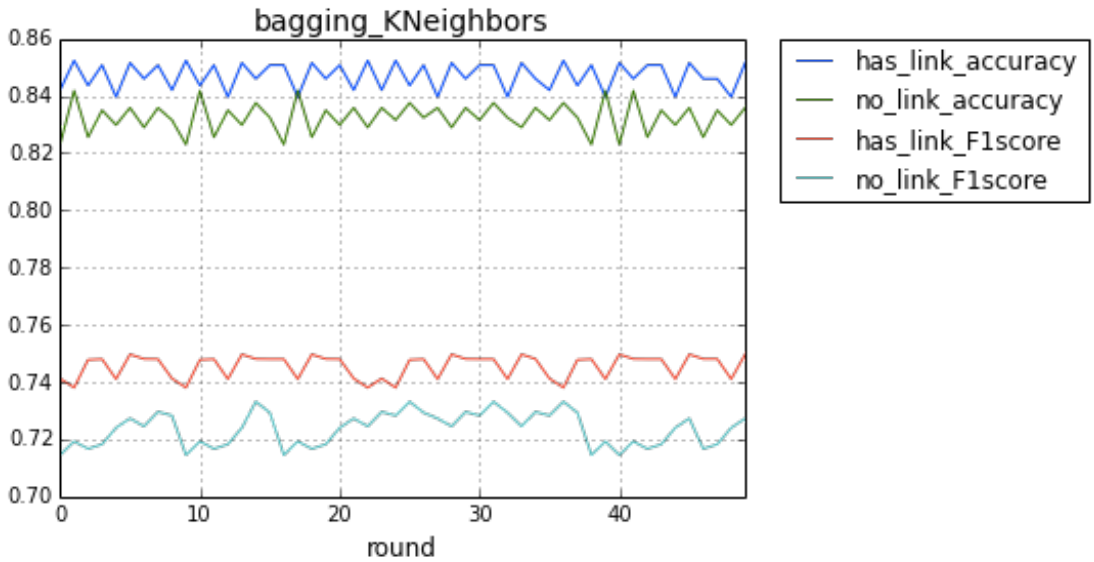
- F1 score

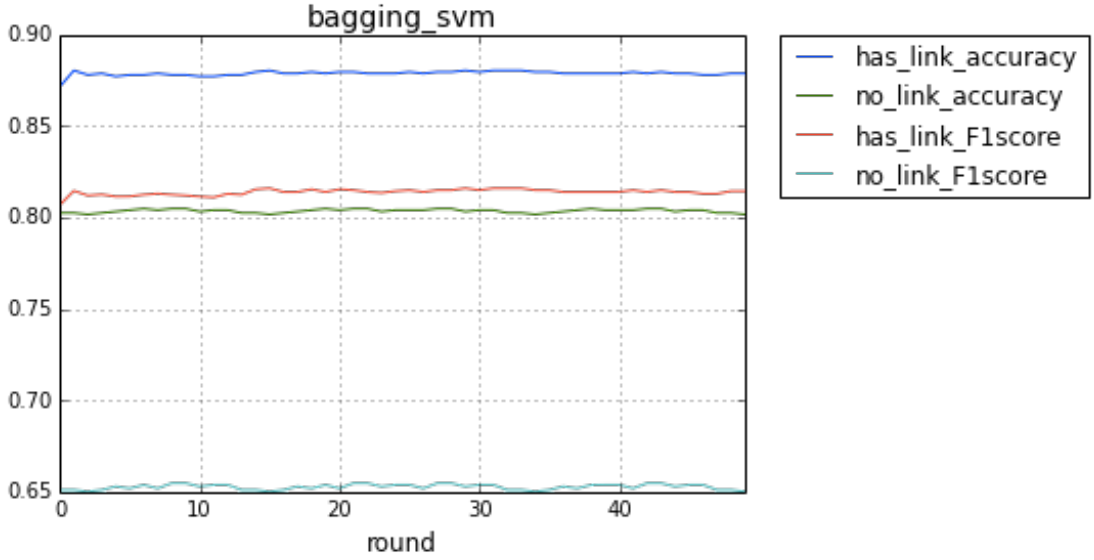


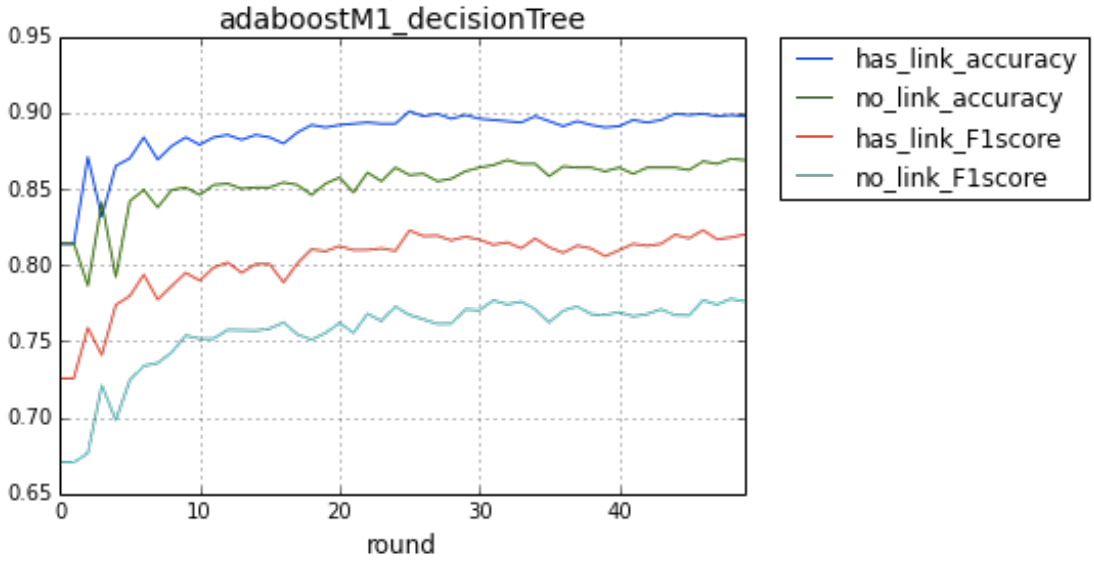
- With/Without Link Feature


bagging_decisionTree
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0
10
20
30
40
round
has_link_accuracy
no_link_accuracy
has_link_F1score
no_link_F1score

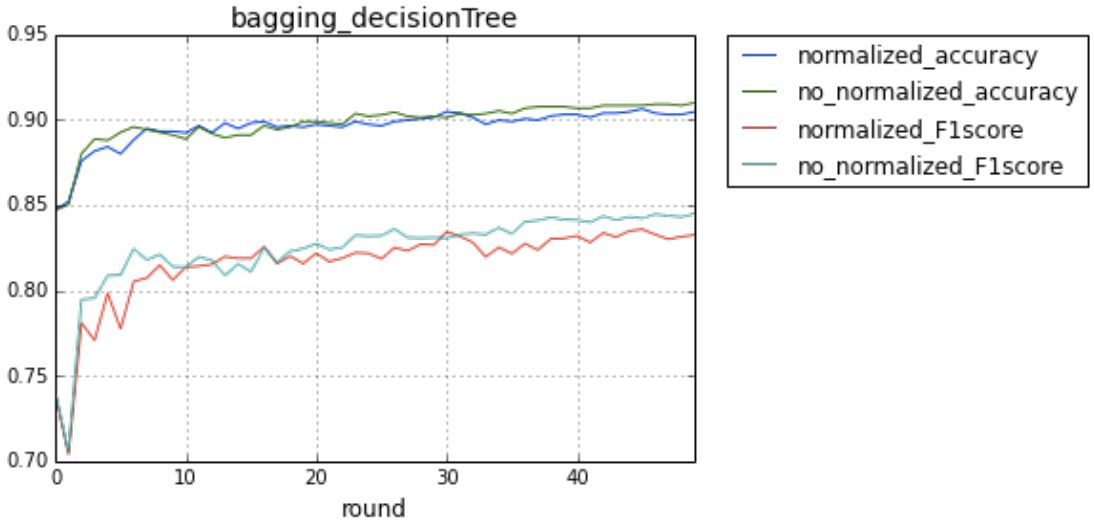

bagging_NaiveBayes
0.92
0.90
0.88
0.86
0.84
0.82
0.80
0.78
0.76
0.74
0
10
20
30
40
round
has_link_accuracy
no_link_accuracy
has_link_F1score
no_link_F1score

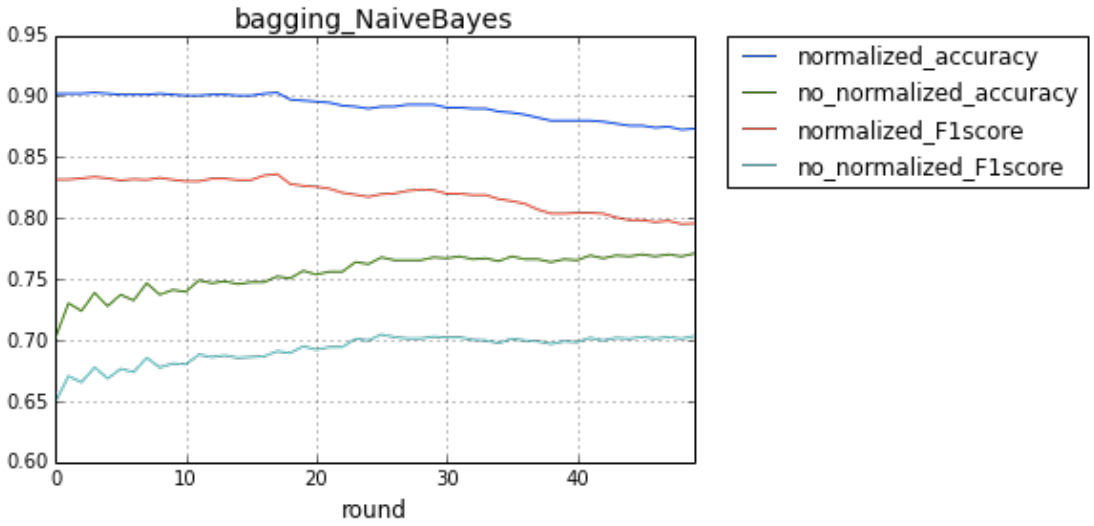

bagging_KNeighbors
0.86
0.84
0.82
0.80
0.78
0.76
0.74
0.72
0.70
0
10
20
30
40
round
has_link_accuracy
no_link_accuracy
has_link_F1score
no_link_F1score

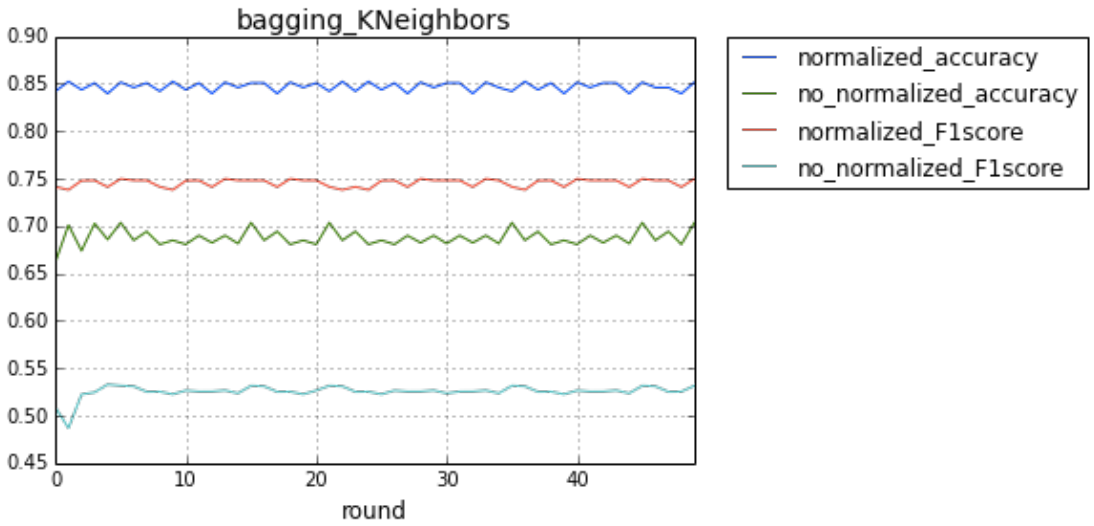


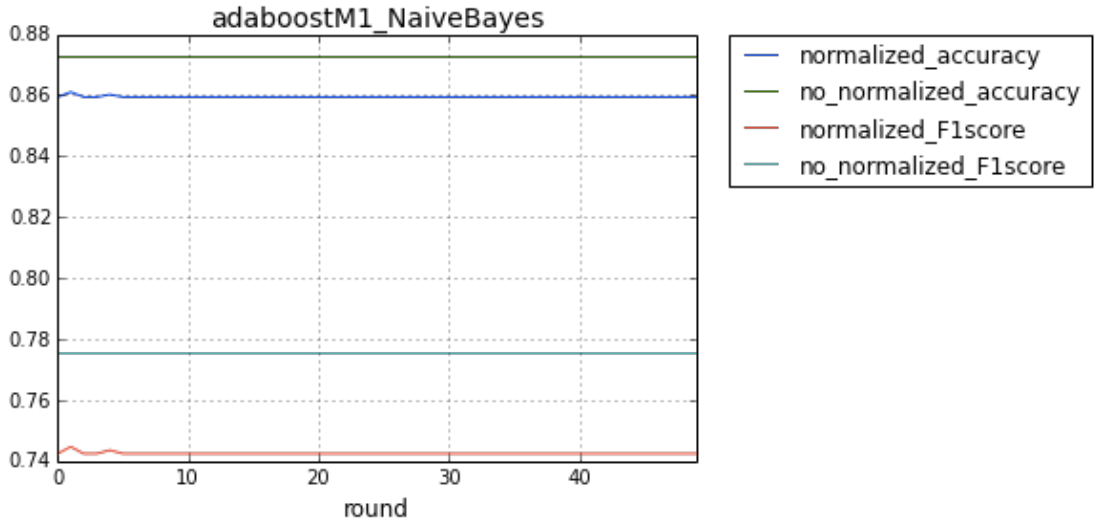


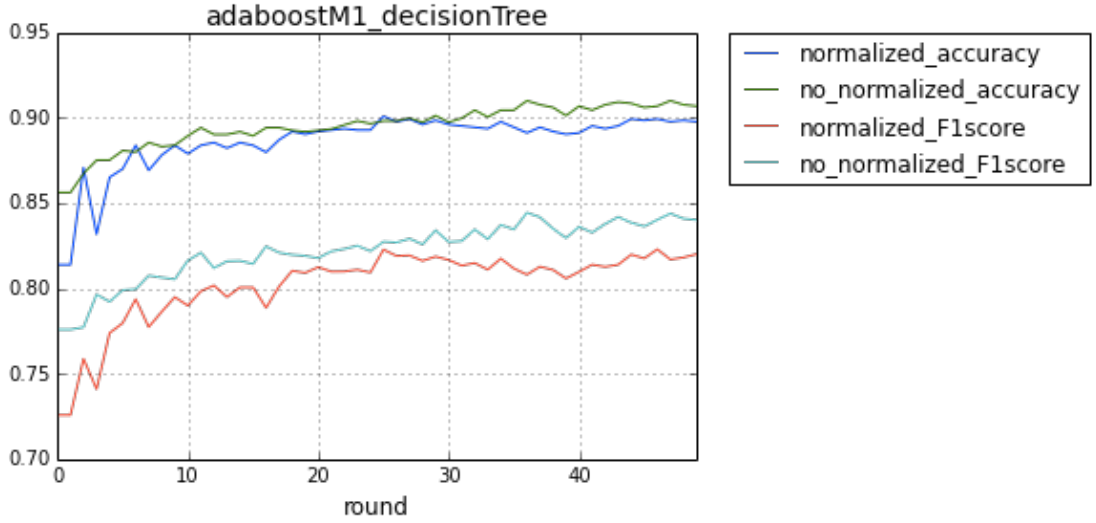
- With/Without Feature Normalization


bagging_decisionTree
0.95
0.90
0.85
0.80
0.75
0.70
0
10
20
30
40
round
normalized_accuracy
no_normalized_accuracy
normalized_F1score
no_normalized_F1score


bagging_NaiveBayes
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0
10
20
30
40
round
normalized_accuracy
no_normalized_accuracy
normalized_F1score
no_normalized_F1score


bagging_KNeighbors
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0.45
0
10
20
30
40
round
normalized_accuracy
no_normalized_accuracy
normalized_F1score
no_normalized_F1score





四、结论和分析

1. Ensemble Learning 的效果

从结果中可以看出，bagging 和 boosting 对于 Decision Tree 这样简单而不稳定的分类器，分类效果有所提升。随着 ensemble 的轮数增加，分类的效果逐渐变好，开始的时候分类效果提升得很快，20 轮之后逐渐趋于平缓。其中效果最好的是 bagging + Decision Tree。在这个 dataset 中 boosting 的表现不如 bagging，可能是因为存在噪声。

但是 boosting 和 bagging 对于 svm 这样稳定而复杂的分类器没有什么提升。对于 Naïve Bayes，bagging 和 boosting 也没有效果上的提升。

对于 KNN（实验中 N=3），bagging 和 boosting 都没有什么作用，因

为对于 bagging 来说，bootstrap 方法中抽取出的样本和待分类样本的最近距离总是差不多的。而对 boosting 来说，不同 sample 的 weight 对 KNN 算法的结果完全没有影响，最终输出的结果每轮都是一样的。

2. With/Without Link Feature

从变化曲线图可以看出，加入了 link feature 后的 Accuracy（蓝线）和 F1 score（红线）都要高于原来的 Accuracy（绿线）和 F1 score（青线）。说明 link feature 对于分类准确率有很大提升

3. With/Without Normalization

从变化曲线图可以看出，normalize 后的 Naïve Bayes、KNeighbors、SVM 的分类 Accuracy 和 F1 score 都更高。但是 Decision Tree 的分类效果并没有明显改进，这是因为 Decision Tree 并不依赖于 feature 的线性组合，feature 的相对大小对其影响不大

五、实验总结

通过这次实验，我对 ensemble learning 有了更加深入的认识，特别是 bagging 和 boosting 的方法。在实验的过程中，通过对比了解了不同的弱分类器对分类效果的影响，了解到 ensemble learning 的精髓在于将一些不稳定的比较“弱”的分类器集成在一起，提高总体分类精度。而对一些本身就很复杂的分类器（如 SVM）就没有太多的提升效果。在实验的过程中，我也掌握了 scikit-learn 的 python 包的使用，收获很大！