Ensemble Learning 实验报告

作者： 陈熠豪

学号：2017011486

邮箱： chenyiha17@mails.tsinghua.edu.cn

代码实现

- data_load.py。实现数据读取和预处理
  - 读取原始数据，将 reviewerID、asin、unixReviewTime 等字段映射为离散数值
  - 调用 gensim，使用训练数据中的 summary、reviewText 内容进行 word2vec 训练
  - 使用 word2vec 模型，把每个词的词向量叠加并和其他特征拼接，得到各个 instance 的特征向量
  - 对预处理数据实现了缓存机制
- bagging.py。实现 bagging 框架
  - 实现 bootstrap，对数据进行随机选取
  - 训练过程与分类器模型解耦，并实现并行化
  - 使用训练好的所有模型进行投票，得到最终结果
- adaboost.py。实现 adaboost.M1 框架
  - 训练过程与分类器模型解耦，实现完全依照 adaboost.M1 算法，此处不再赘述
  - 使用训练好的所有模型进行投票，得到最终结果
- classifier.py。实现具体的 base classifier model
  - svm。调用 sklearn.svm.LinearSVC，由于训练数据量过大，根据 sklearn 文档建议，训练参数中的 dual 置为 False
  - dt。调用 sklearn.tree.DecisionTreeClassifier，为避免过拟合，训练参数中的 max_depth 建议不超过 10
- 其他代码
  - main.py 主函数，实现整体的实验调用逻辑
  - ./src/cnn/下的代码为 cnn 分类器的训练代码，但由于训练难度大和神经网络本身的迭代特性，没有纳入到集成学习的框架中，仅仅作为对照和参考，此处不作赘述

实验结果

集成框架	迭代次数	分类器模型	词向量维度	附加参数	RMSE
adaboost	20	dt	300	dt 最大深度 5	1.35449
adaboost	20	dt	300	dt 最大深度 10	1.28957
adaboost	20	dt	300	dt 最大深度 20	1.28802

集成框架	迭代次数	分类器模型	词向量维度	附加参数	RMSE
adaboost	20	svm	300	-	1.35425
bagging	10	dt	300	dt 最大深度 5	1.32060
bagging	10	dt	300	dt 最大深度 10	1.23827
bagging	10	dt	300	dt 最大深度 20	1.41185
bagging	10	svm	300	-	1.35720
-	-	cnn	300	使用 summary 数据	1.00813
-	-	cnn	300	使用 reviewText 数据	1.07641

注:

- 上表中"迭代次数"指集成学习框架中的迭代次数（即最终的分类器模型个数）
- 在集成学习的实验中，除词向量外，还有其他字段的维度，比如 reviewerID、asin、unixReviewTime，因此实际的特征向量维度是 303

分析

- svm 分类器不能很好地处理本次实验的数据，因为此次实验的数据量较大，不得不采用线性 svm，而且为了降维，我使用 word2vec 来做词向量，因此最后的特征会变得比较复杂，线性 svm 不能很好地实现分类。采用 bagging + svm 的话，相当于对不同的分布做了平滑处理，是可以对分类效果有一定提升，但从本质上来讲，这仅仅是在输入层面上的优化，由于 svm 本身的算法特点，其实无法在整体上带来太大的提升，而成倍增加的时间和空间开销也是不能忽略的因素。采用 adaboost + svm 的话，由于 svm 分类能力过弱，迭代收敛极快，在第二个周期就退出了 adaboost 的训练过程，因此最终的结果并没有很好地提升。

- dt 分类器的训练较为简单，但是需要处理过拟合的问题。经过实践发现，采用 bagging + dt (depth=10) 能取得不错的结果，一方面是因为限制了深度使得 dt 不易过拟合，另一方面 dt 本身是数据敏感的算法，因此 bagging 能够较好地带来提升。adaboost + dt (depth=20)也有很好的提升，我觉得是因为 adaboost 从输入层面上进行了反馈调节，在某种程度上缓解了 dt (depth=20)的过拟合问题。

- adaboost 和 bagging 之间的区别：bagging 相当于将数据分布进行了平滑处理，使得数据敏感的分类器的综合效果得以提升，并且可以很方便的并行化；adaboost 是在数据层面上进行了反馈调节，使得训练目标可以沿着确定的方向迁移，但是对 base classifier 有一定的要求，否则可能在迭代时失效，而且这种迭代过程是难以并行化的。

- 这两种集成学习方法虽然有一定的效果，但我觉得本质上还是受 base classifier 的局限的，只有改变 base classifier 的算法才能有显著的优化。使用神经网络可以更好地处理复杂的分类问题，比如简单的单层 cnn 就可以轻松地获得更理想的结果。虽然神经网络分类器不是不能够加入到集成学习框架中，但我觉得训练成本太高，与其进行这样的横向优化，不如提升神经网络的深度更加方便。
