数据挖掘导论
作业三

151250189, 翟道京, zhaidj@smail.nju.edu.cn

2018 年 6 月 1 日

摘要

在本次数据挖掘任务中，我们使用 Weka 软件，对所提供的十组数据集进行多种分类器的训练，并集中比较了六种学习方式的优异性；文末我们对基于 CNN 方法的 Bagging 集成学习进行了性能优化，通过相关参数修改，提升了对 german_credit 数据集的分类功能。

1 任务描述

Try J4.8 (C4.5), Naïve Bayes, SVM, Neural Network, kNN, and their ensemble version using Bagging on provided data sets based on 10-fold cross validation, can compare their performances w.r.t, accuracy, and AUC. Discuss on their performance and suggest how to improve Bagging of KNN (with necessary experimental evidence).

2 数据描述

本次数据挖掘实验采用十组数据集如下所列

- wisconsin-breast-cancer
- horse-colic
- credit-rating
- german_credit
- pima_diabetes
- hepatitis
- mozilla4
- PC1
- pc5

- waveform

十组数据集样本维度各不同，均为二分类任务。

3 实验方法

本次作业选用 Weka 软件实现 J4.8 (C4.5), Naïve Bayes, SVM, Neural Network, kNN 与 Bagging 功能，使用如下内建函数

- weka.classifiers.trees.J48
- weka.classifiers.bayes.NaiveBayes
- weka.classifiers.functions.SMO
- weka.classifiers.functions.MultilayerPerceptron
- weka.classifiers.lazy.IBK
- weka.classifiers.meta.Bagging

各函数参数均选择初始默认参数设置。其中 Bagging 基于 REPTree 算法。使用 10 次交叉验证，通过 Weka 软件对相关训练器的训练，我们获得相应的学习性能指标[1]。

4 实验结果

4.1 wisconsin-breast-cancer Data Set

表 1: wisconsin-breast-cancer 数据集训练结果

Method	Accuracy	Recall	ROC Area	PRC Area	Time/s
J4.8(C4.5)	0.946	0.946	0.955	0.937	0.06
Naïve Bayes	0.962	0.960	0.986	0.976	0.02
SVM(SMO)	0.970	0.970	0.968	0.957	0.05
Neural Network	0.953	0.953	0.896	0.984	0.64
1-NN	0.954	0.954	0.988	0.986	0.00
Bagging	0.965	0.964	0.990	0.987	0.16

[1]注意 Weka 显示的时间为模型构建时间，而非训练时间。

4.2 horse-colic Data Set

表 2: horse-colic 数据集训练结果

Method	Accuracy	Recall	ROC Area	PRC Area	Time/s
J4.8(C4.5)	0.854	0.853	0.813	0.788	0.02
Naïve Bayes	0.788	0.780	0.842	0.843	0.01
SVM(SMO)	0.825	0.826	0.809	0.769	0.17
Neural Network	0.806	0.804	0.857	0.848	4.67
1-NN	0.811	0.813	0.802	0.766	0.00
Bagging	0.864	0.864	0.893	0.892	0.16

4.3 credit-rating Data Set

表 3: credit-rating 数据集训练结果

Method	Accuracy	Recall	ROC Area	PRC Area	Time/s
J4.8(C4.5)	0.861	0.861	0.887	0.849	0.01
Naïve Bayes	0.793	0.777	0.896	0.886	0.00
SVM(SMO)	0.861	0.849	0.856	0.806	0.25
Neural Network	0.836	0.836	0.895	0.885	4.52
1-NN	0.811	0.812	0.808	0.753	0.00
Bagging	0.860	0.857	0.919	0.908	0.04

4.4 german-credit Data Set

表 4: german-credit 数据集训练结果

Method	Accuracy	Recall	ROC Area	PRC Area	Time/s
J4.8(C4.5)	0.687	0.705	0.639	0.657	0.02
Naïve Bayes	0.743	0.754	0.787	0.797	0.00
SVM(SMO)	0.738	0.751	0.671	0.681	0.29
Neural Network	0.713	0.715	0.730	0.757	12.36
1-NN	0.716	0.720	0.660	0.669	0.00
Bagging	0.732	0.747	0.762	0.773	0.06

4.5 pima-diabetes Data Set

表 5: pima-diabetes 数据集训练结果

Method	Accuracy	Recall	ROC Area	PRC Area	Time/s
J4.8(C4.5)	0.735	0.738	0.751	0.727	0.01
Naïve Bayes	0.759	0.763	0.819	0.815	0.00
SVM(SMO)	0.769	0.773	0.720	0.698	0.03
Neural Network	0.750	0.754	0.793	0.786	0.50
1-NN	0.696	0.702	0.650	0.640	0.00
Bagging	0.752	0.758	0.812	0.808	0.03

4.6 hepatitis Data Set

表 6: hepatitis 数据集训练结果

Method	Accuracy	Recall	ROC Area	PRC Area	Time/s
J4.8(C4.5)	0.825	0.839	0.708	0.800	0.00
Naïve Bayes	0.853	0.845	0.860	0.891	0.00
SVM(SMO)	0.847	0.852	0.756	0.803	0.01
Neural Network	0.807	0.800	0.823	0.848	0.28
1-NN	0.794	0.806	0.653	0.747	0.00
Bagging	0.786	0.813	0.800	0.829	0.01

4.7 mozilla4 Data Set

表 7: mozilla4 数据集训练结果

Method	Accuracy	Recall	ROC Area	PRC Area	Time/s
J4.8(C4.5)	0.949	0.948	0.954	0.953	0.42
Naïve Bayes	0.785	0.686	0.829	0.824	0.02
SVM(SMO)	0.848	0.832	0.838	0.800	2.65
Neural Network	0.911	0.912	0.940	0.944	5.88
1-NN	0.890	0.890	0.877	0.857	0.01
Bagging	0.951	0.950	0.975	0.979	0.82

4.8 pc1 Data Set

表 8: pc1 数据集训练结果

Method	Accuracy	Recall	ROC Area	PRC Area	Time/s
J4.8(C4.5)	0.917	0.933	0.668	0.901	0.02
Naïve Bayes	0.899	0.892	0.650	0.900	0.00
SVM(SMO)	0.866	0.930	0.500	0.871	0.05
Neural Network	0.921	0.936	0.723	0.916	2.19
1-NN	0.922	0.921	0.740	0.918	0.00
Bagging	0.931	0.941	0.847	0.943	0.06

4.9 pc5 Data Set

表 9: pc5 数据集训练结果

Method	Accuracy	Recall	ROC Area	PRC Area	Time/s
J4.8(C4.5)	0.972	0.975	0.817	0.967	1.72
Naïve Bayes	0.966	0.964	0.833	0.971	0.12
SVM(SMO)	0.966	0.972	0.541	0.946	12.68
Neural Network	0.966	0.971	0.941	0.981	101.63
1-NN	0.972	0.973	0.932	0.979	0.01
Bagging	0.972	0.976	0.975	0.987	2.57

4.10 waveform Data Set

表 10: waveform 数据集训练结果

Method	Accuracy	Recall	ROC Area	PRC Area	Time/s
J4.8(C4.5)	0.751	0.751	0.830	0.672	0.33
Naïve Bayes	0.835	0.800	0.956	0.919	0.04
SVM(SMO)	0.867	0.867	0.932	0.817	0.28
Neural Network	0.836	0.836	0.963	0.929	34.03
1-NN	0.736	0.736	0.802	0.630	0.00
Bagging	0.815	0.815	0.951	0.903	1.04

5 结果分析

根据上述数据集的结果分析，得到以下结论

1. 简单的 Mutilayer Perceptron(CNN) 在不同训练集上都有着不俗的表现，但同时缺陷为最大的时间开销。

2. 就时间开销而言，KNN 作为典型的 lazy learning 方法，时间开销较小；同时 Naïve Bayes 方法也有这较优秀的时间开销。

3. 就分类性能而言，对于不同的问题，判别式模型与生成式模型有着不同的表现，对于具体问题，应当在对数据特征有着深刻理解后，选择相应的学习方法。

4. 基于 REPTree 的简单集成学习 Bagging 方法对不同的问题都有着不俗的表现，展现出集成学习较为强大的泛化能力。

6 算法优化

我们上述实验中的集成学习是基于 REPTree 进行 Bagging，其中对部分数据集效果并不出众，我们以 german_credit 数据集为例，探究提升 Bagging of KNN 的性能的方法。

6.1 KNN 属性的探究

通过调节 K-NN 属性 K 值，对分类器性能进行提升，下表展示了单层 Bagging 下不同 K 值的分类器性能

表 11: My caption

K-NN	Precision	Recall	ROC Area	PRC Area	Time/s
1-NN	0.713	0.721	0.694	0.722	0.01
3-NN	0.715	0.732	0.721	0.754	0.01
5-NN	0.716	0.735	0.743	0.776	0.01
10-NN	0.717	0.737	0.755	0.777	0.01

由上表可知，通过对本数据集的研究，调节 K 值对分类器性能有一定的提升。

6.2 Bagging 迭代次数的研究

我们选择 10-NN 分类器，在单层 Bagging 下调节不同迭代次数，探究分类器的性能变化

表 12: My caption

Iteration	Precision	Recall	ROC Area	PRC Area	Time/s
10	0.717	0.737	0.755	0.777	0.01
20	0.730	0.747	0.753	0.779	0.01
50	0.724	0.742	0.753	0.780	0.03
100	0.727	0.744	0.755	0.782	0.03

如上表所示，在一定范围内调节 Bagging 迭代次数，可提高分类器性能，但迭代次数超过一定范围后，提升效果不再显著。

6.3 Bagging 层嵌套

我们选择 Bagging 迭代次数为 10 次，在基于 10-NN 分类标准下实现 Bagging 层的嵌套，从而提升分类器的性能

表 13: My caption

Layers	Precision	Recall	ROC Area	PRC Area	Time/s
1	0.717	0.737	0.755	0.777	0.01
2	0.726	0.743	0.761	0.787	0.05
3	0.728	0.744	0.762	0.788	0.29
4	0.747	0.746	0.788	0.793	4.38
5	0.776	0.767	0.800	0.813	106.40

可以发现，尽管牺牲了时间开销，多层 Bagging 嵌套分类器性能得到了提升。

7 总结

在本次实验中，我们使用 Weka 程序演绎了经典的几种机器学习分类方法，对相关方法性能进行评估比较，并通过调参实验，深入探讨了 KNN-Bagging 集成学习方法。本次实验后我又以下心得体会：

- 具体问题具体分析，对于不同的数据，我们应该在了解相应数据特征、分布类型后选择对应的学习方法。
- 不同的学习方法的时间开销/空间开销中差异较大，有着特定的适用问题，我们需要了然于胸。
- 掌握必要的调参方式，在机器学习，特别是深度学习中有着重要意义。