Project3
Classification Algorithms

Jingsong Li	Junyang Li	Kaige Gao
50322345	50320301	50320915
jingsong@buffalo.edu	*junyangl@buffalo.edu*	*kaigegao@buffalo.edu*

1 Introduction

The task of this project is to implement three classification algorithms: Nearest Neighbor, Decision Tree, and Naïve Bayes. And Implement Random Forests based on the implementation of Decision Tree. Then adopt 10-fold Cross Validation to evaluate the performance of all methods in terms of accuracy, precision, recall, and F-1 measure. Finally, submit a classification result of a kaggle competition.

2 Dataset

Two datasets will be used as the example data in this project.

3 Preprocess

3.1 Read data

With the given text file, the program will read each line of it and store as a two-dimension list.

3.2 Data cleaning

The last column of the datasets we get is the label of the sample. So the program will separate the columns as the label(last column) and attributes(other columns).

3.2 String converting

The values of one attribute in dataset2 are string ‘absent’ or string ‘present’. We cannot directly process this attribute, so we need to convert it first.

For string attributes having three different possible values, we keep them in the list and use equal or not equal to separate them into two groups when separating the groups. It’s tricky because with string attributes having possible value larger than three, this way won’t work, only combination can separate them without any missing possibility. Luckily, this time we still can use equal or not equal to solve the question.

4 Implementations

4.1 Nearest Neighbor

The first step is to compute the distance from one record to other training records.

The second step is to identify k nearest neighbors.

The last step is to use class labels of nearest neighbors to determine the class label of unknown records by taking the majority vote.

4.2 Decision Tree

The main iteration is to choose the node to divide and divide the data set with the node chosen. The first part is to choose each node to calculate the gini index to determine whether to choose it to be the divide node. We want the node that having the smallest gini index to be the divide node. For continues variables, we can simply compare whether it's larger than the node value chosen; but for distant values, we can use the same way if it has only two classes, for string attributes, it's still possible to use equal or not equal to separate them into two groups without missing any possibility. In this program, we use the latter way to deal with the distant values. The second part will divide the data set into two parts and decide whether to continue choose the next divide node.

For prediction, it's easy to follow the rule to choose which child to go and finally arrive the leaf node and predict the class it belongs to.

4.3 Naïve Bayes

Continuous data handling is achieved similarly by first storing the data as a list of lists and then casting and extracting it to numpy array. Given that the continuous data have numerical values, numpy is extremely useful and adept at handling these kind of data. We're using Probability Distribution Function to calculate posterior probabilities.

Zero-probability - Posterior probability goes to 0 if any of probability is 0. To correct this we've a check in place that corrects this problem by adding 1 to each case - this is also known as Laplacian correction (or Laplacian estimator).

4.4 Random Forests

Similar to decision tree, the main iteration steps are same, the difference is at random forests will choose several attributes to do the decision, which can save a lot of time. In order to avoid losing important attribute, random forest will generate multiple trees to choose the one with the best result.

5 Results

All evaluation results of dataset1 and dataset2 are averaged based on 10-fold cross-validation.

5.1 Nearest Neighbor

dataset1

k	10	20	30	40	50
accuracy	0.93571	0.93571	0.92679	0.92143	0.91786
precision	0.93510	0.95323	0.95169	0.94795	0.96144
recall	0.88907	0.86885	0.84202	0.82906	0.82241
F-1 measure	0.90967	0.90748	0.89064	0.88029	0.88415

dataset2

k	10	20	30	40	50
accuracy	0.62826	0.66957	0.66087	0.66304	0.68478
precision	0.43962	0.55561	0.59000	0.48418	0.62393
recall	0.28005	0.29036	0.20022	0.23561	0.24843
F-1 measure	0.33459	0.35734	0.28666	0.31177	0.34211

5.2 Decision Tree

dataset1

accuracy	0.9182467532467534
precision	0.896021493844217
recall	0.8880746150057783
F-1 measure	0.8904360479184467

dataset2

accuracy	0.6606763285024154
precision	0.5502610532338963
recall	0.4353989934703922
F-1 measure	0.468850438642218

5.3 Naïve Bayes

dataset1

accuracy	0.8240601503759398
precision	0.8240601503759398
recall	0.9670250582750584

F-1 measure	0.8034977281420052

dataset2

accuracy	0.6816373728029602
precision	0.6816373728029602
recall	0.5728314659197012
F-1 measure	0.5464670986639927

dataset4

accuracy	0.6
precision	0.6
recall	0.7
F-1 measure	0.6333333333333333

5.4 Random Forests

dataset1

accuracy	0.9457792207792208
precision	0.9594194071801875
recall	0.8959855591618225
F-1 measure	0.9243250846481894

dataset2

accuracy	0.7276328502415459
precision	0.7096283642137134
recall	0.3643008525625571
F-1 measure	0.42970206364001234

6 Evaluation

6.1 Nearest Neighbor

Advantages

1. It is relatively simple, easy to understand and implement, high precision and mature theory.
2. Can be used for non-linear classification.

3. The run time of training is O(n) and relatively low.
4. Not sensitive to outliers.

Disadvantages

1. Large amount of calculation, especially when the number of features is very large.
2. High spatial complexity.
3. When the samples are unbalanced, the prediction accuracy for rare categories is low.
4. k-NN classifiers are lazy learners. Classifying unknown records are relatively expensive.

6.2 Decision Tree

Advantages

1. Inexpensive to construct.
2. Extremely fast at classifying unknown records.
3. Easy to interpret for small-sized trees.
4. Accuracy is comparable to other classification techniques for many simple data sets.

Disadvantages

1. Hard to deal with nominal attributes.
2. Have a risk of overfitting.

6.3 Naïve Bayes

Advantages

1. It is fairly easy to implement Naive Bayes classifier.
2. The technique requires a small amount of training data to estimate the parameters.
3. The implementation is highly scalable and scales linearly with the number of predictors and data points.
4. The classifier works for both the binary and multiclass features.

Disadvantages

1. A very strong assumption on the data distribution i.e. features being completely independent given their label class can lead to less accuracy. Hence “naive” classifier is just as what it named.
2. Continuous features need a special handling, usually Gaussian based distribution functions works well to adapt the classifier to continuous data. Though techniques like binning can them discrete but also lose useful information if not careful.

6.4 Random Forests

Advantages

1. Training can be very efficient. Particularly true for very large datasets.
2. Natural multi-class probability.
3. Imposes very little about the structures of the model.

Disadvantages

1. May miss important features.
2. Cost more resource when generating multiple tree.

7 Conclusion

Based on the above, the classification result of dataset1 is obviously better than dataset2.

8 Kaggle

We applied 8 classification algorithms (including nearest neighbor, decision tree, Naïve Bayes, SVM, logistic regression, bagging, AdaBoost, random forests) and tune parameters using training dataset. And used the trained model to classify the data in the test dataset and evaluate the results. Among all the results, AdaBoost algorithm has the best classification result.

In the process of parameter tuning of the AdaBoost algorithm, a better result can be obtained when n_estimators(Number of base classifier promotions (cycles)) equal to 750. If it is less than this value, it will underfitting, and if it is greater than this value, then will overfitting. The score of our results is 0.85915.