International Journal of Applied Research

ISSN Print: 2394-7500
ISSN Online: 2394-5869
Impact Factor: 5.2
IJAR 2017; 3(6): 174-178
www.allresearchjournal.com
Received: 25-04-2017
Accepted: 26-05-2017

Aranga Arivarasan
Assistant Professor
Programmer, Department of
Computer and Information
Science Annamalai University
Tamilnadu, India

Dr. M karthikeyan
Assistant Professor
Programmer, Department of
Computer and Information
Science Annamalai University
Tamilnadu, India

Correspondence
Dr. Mallika Kalita
Assistant Professor
Programmer, Department of
Computer and Information
Science Annamalai University
Tamilnadu, India

Classification based Performance analysis using Naïve-Bayes J48 and Random forest algorithms

Aranga Arivarasan and Dr. M karthikeyan

Abstract
Classification is an important data mining task with broad applications to classify the various kinds of data used in nearly every field in our day to day life. Classification is used to classify each item according to the features of the item with respect to the predefined set of classes or groups. In this paper we focus on the performance evaluation based on the correct and incorrect instances of data classification using Naïve-Bayes, J48 and Random Forest classification algorithms. Naïve-Bayes algorithm is based on probability. J48 algorithm is based on decision tree and Random Forest is a way of averaging multiple deep decision trees. This work deals with comparative evaluation of classifiers NAÏVE-BAYES, J48 AND RANDOMFOREST in the context of dataset to maximize true positive rate and minimize false positive rate using WEKA tool. Experimental results shows that the Random forest algorithm achieves an accuracy of 94.50% for the CAR dataset

Keywords: True positive rate, false positive rate, Precision, Recall, F-measure

Introduction

Data mining is an automated process of discovering interesting information or patterns by means of understandable predictive models from large data sets by grouping them. Data mining is widely growing in various applications like analysis of organic compounds, medicals diagnosis, product design, predictive analysis in marketing, fraud detection in credit card, financial forecasting, automatic abstraction, predicting the instability of share marketing etc. Data mining is applied in many type of media or data. Data mining is applicable to any kind of information repository. Data mining is uses to study about the knowledge hidden inside the databases, like relational databases, object-relational databases and object oriented databases, data warehouses, transactional databases, unstructured and semi-structured WWW, multimedia databases, time-series databases and textual databases.

Different functions of data mining are feature selection, classification, clustering and association rule mining. In this work, we focus on the data classification and the performance achieved by the classifier algorithms through True Positive rate, False Positive rate by the Naïve-Bayes, J48 and Random Forest when applied on the data set. Classification analysis through the organization of data in given predefined set of classes. The classification uses given class labels to order the objects in the data collection. Classification approaches normally use a training set where all objects are already associated with known class labels. The classification algorithm learns from the training set and builds a model for testing set. The model is used to classify new objects. In our experiment we use the uci machine learning repository car. arff data set which is derived from simple hierarchical decision model. This dataset is very much useful for testing constructive induction and structure discovery methods using various data mining algorithms. The Database Car Evaluation contains examples with a removal of structural information.

2. Classifiers

2.1 Naïve-Bayes

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

Naïve-Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity. Naïve-Bayes is known to outperform even highly sophisticated classification methods. Naïve-Bayes is a simple technique for constructing classifier models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle all Naïve-Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For some types of probability models, Naïve-Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for Naïve-Bayes models uses the method of maximum likelihood, in other words, one can work with the Naïve-Bayes model without accepting Bayesian probability or using any Bayesian methods.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c).

$$Posterior = \frac{Prior\ X\ likelihood}{evidence}$$

Naïve-Bayes is a conditional probability model given a problem instance to be classified, represented by a vector $X = (x_1, .. x_n)$ representing some n features (independent variables), it assigns to this instance probabilities $P(c_k|x_1, \ldots, x_n)$ for each of K possible outcomes or classes

2.2 J48 Algorithm

J48 classifier is a simple C4.5 decision tree for classification. It creates a binary tree. The decision tree approach is most useful in classification problem. With this technique, a tree is constructed to model the classification process. Once the tree is built, it is applied to each tuple in the database and results in classification for that tuple.

Tree Algorithm is to find out the way the attributes-vector behaves for a number of instances found inside the database. This algorithm generates the rules for the prediction of the target variable. With the help of tree classification algorithm the critical distribution of the data is easily made understandable.

2.2.1 Algorithm for Decision Tree Induction

Basic algorithm (a greedy algorithm) Tree is constructed in a top-down recursive divide-and conquer Manner

1. At start, all the training examples are at the root.
2. Attributes are categorical (if continuous-valued, they are discredited in advance)
3. Examples are partitioned recursively based on selected attributes
4. Test attributes are selected on the basis of a heuristic or statistical measure.

Condition for stopping partitioning

1. All samples for a given node belong to the same class
2. There are no remaining attributes for further partitioning- majority voting is employed for classifying the leaf
3. There are no samples left

In the WEKA data mining tool, for J48 is an open source Java implementation of the C4.5 algorithm. The WEKA tool provides a number of options associated with tree pruning. In other algorithms the classification is performed recursively till every single leaf is pure, that is the classification of the data should be as perfect as possible.

2.3 Random forests

Random Forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error for forests converges to a limit as the number of trees in the forest becomes large. Random Forests is a special kind of ensemble learning technique and robust concerning the noise and the number of attributes. Random Forests are an ensemble learning method for classification that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes output by individual trees. In a Random Forests, each node is split using the best among the subset of predicators randomly chosen at that node. To classify a new object from an input vector, the algorithm put the input vector down to each of the trees in the forest. Each tree gives a classification, and we say the tree "votes" for that class. The forest chooses the classification having the most votes.

Each tree is grown as follows:

1. The number of cases in the training set is N, sample N cases at random - but *with replacement*, from the original data. This sample will be the training set for growing the tree.
2. If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.
3. Each tree is grown to the largest extent possible. There is no pruning.
4. In original the random forests shows that the forest error rate depends on two things:

- The *correlation* between any two trees in the forest. Increasing the correlation increases the forest error rate.
- The *strength* of each individual tree in the forest. A tree with a low error rate is a strong classifier. Increasing the strength of the individual trees decreases the forest error rate.

Reducing m reduces both the correlation and the strength. Increasing it increases both. Somewhere in between is an "optimal" range of m - usually quite wide. Using the error rate a value of m in the range can quickly be found. This is the only adjustable parameter to which Random Forests is somewhat sensitive. Often for every training s labels assignments are formed by human judgment. In some areas this leads to a high frequency of mislabeling. Many of the mislabeled cases can be detected using the outlier measure in Random Forests.

3. System overview

The performance of classification algorithm is usually examined by evaluating the accuracy of the classification. The correct answer may depend on the user because the classification is a fuzzy problem. Classification accuracy is usually calculated by determining the percentage of tuples placed in a correct class. This ignores the fact that there also may be a cost associated with an incorrect assignment to the wrong class. This perhaps should also determine.

An OC (operating characteristics) curve or ROC (receiver operating characteristic) curve or ROC (relative operating characteristic) curve shows the relationship between false positives and true positives. An OC curve was originally used in communication area examined false alarm rates. It has also been used in information retrieval to examine the percentage of retrieved instances that are not relevant VS percentage of retrieved instances that are relevant.

3.1 Confusion matrix

A confusion matrix is a table that is often used to describe the performance of a classification model or "classifier" on a set of test data for which the true values are known. The confusion matrix itself is relatively simple to understand, but the related terminology can be confusing. It contains information about actual and predicted classifications done by a classification system. Performance of such systems is commonly evaluated using the data in the matrix known as confusion matrix. A confusion matrix of a binary classification is a two by two table counting of the number of the four outcomes of a binary classifier.

Table 1: sample confusion matrix

N=Number of instances	Predicted: YES	Predicted: NO
Actual: YES	TP	FN
Actual: NO	FP	TN

There are two possible predicted classes: "yes" and "no". If we were predicting the presence of an instance in that category, for example, "yes" would mean the presence of the instance of that category, and "no" would mean the absence of the instance of that category. A binary classifier predicts all data instances of a test dataset as either positive or negative. Possibly the classification normally produces only four outcomes as true positive, true negative, false positive and false negative.

3.2 Standards and terms

- **True Positives (TP):** These are cases in which we predicted yes, and the word is present.
- **True Negatives (TN):** We predicted no, and the word is not present.
- **False Positives (FP):** We predicted yes, but the word don't actually present. (Also known as a "Type I error.")
- **False Negatives (FN):** We predicted no, but the word actually do present. (Also known as a "Type II error.")

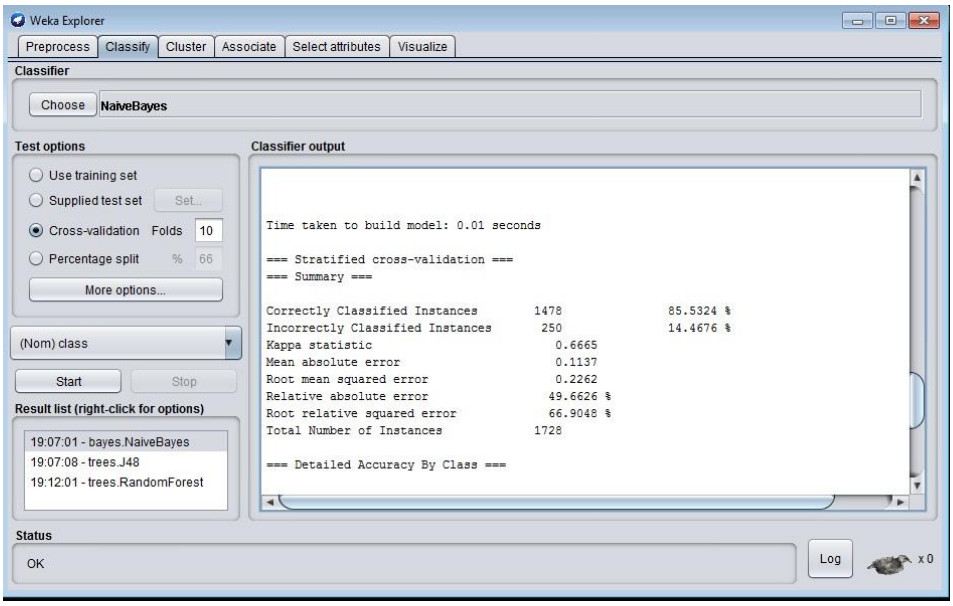
3.3 Calculating the values

- **Accuracy:** Overall, how often classifier is correct (TP+TN)/total
- **True Positive Rate:** When it is actually yes, how often does it predict yes TP/actual yes also known as "Sensitivity" or "Recall"
- **False Positive Rate:** When it is actually no, how often does it predict yes FP/actual no
- **Precision:** When it predicts yes, how often is it correct TP/predicted yes
- **F-measure:** 2. $(Precision.recall)/(Precision + recall)$

4. Results and discussion

4.1 Results using Naïve-Bayes

Here Class attribute has been chosen from car data set. Naïve-Bayes is applied on the data set and the Fig-1 results are achieved.



Classifier: NaiveBayes

Test options: ○ Use training set ○ Supplied test set ● Cross-validation Folds 10 ○ Percentage split % 66

(Nom) class

Result list:
19:07:01 - bayes.NaiveBayes
19:07:08 - trees.J48
19:12:01 - trees.RandomForest

```
Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1478               85.5324 %
Incorrectly Classified Instances       250               14.4676 %
Kappa statistic                          0.6665
Mean absolute error                      0.1137
Root mean squared error                  0.2262
Relative absolute error                 49.6626 %
Root relative squared error             66.9048 %
Total Number of Instances             1728

=== Detailed Accuracy By Class ===
```

Status: OK

Fig 1: WEKA Naïve-bayes classification.

Hence we calculate from Table-1 confusion matrix are:
TP rate for class a = 1161/ (1161+49) = 0.960
FP rate for class a = 104/ (104+414) = 0.203
TP rate for class b = 0.706
FP rate for class b = 0.098
TP rate for class c = 0.275
FP rate for class c = 0.007
TP rate for class d = 0.415
FP rate for class d = 0.001

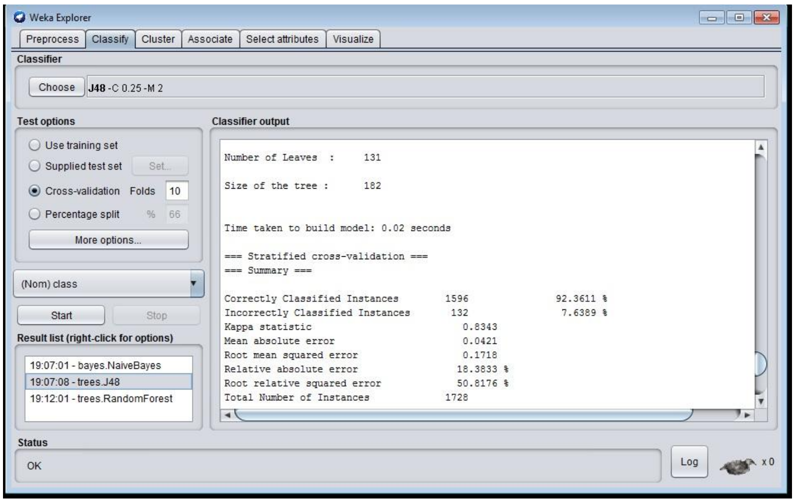
Table 1: Naïve-Bayes classifier confusion matrix

N=1728	Predicted: YES	Predicted: NO
Actual: YES	1161 (True positive)	49 (False Negative)
Actual: NO	104 (False Positive)	414 (Negative for unacc)

Average TP rate = 0.855
Average FP rate = 0.164
Precision for class a = 0.917
Precision for class b = 0.672
Precision for class c = 0.633
Precision for class d = 0.931
F-measure for class a = 2*precision*recall/(precision + recall) = 0.938
F-measure for class b = 0.689
F-measure for class c = 0.384
F-measure for class b = 0.574

4.2 Results using J-48 Here Class attribute has been chosen from car data set. J48 is applied on the data set and the Fig-2 results are being achieved.

Hence we calculate from the Table-2 confusion matrix are:
TP rate for class a = 1164/(1164+46) = 0.962
FP rate for class a = 33/(33+485) = 0.064
TP rate for class b = 0.867
FP rate for class b = 0.047
TP rate for class c = 0.609
FP rate for class c = 0.011
TP rate for class d = 0.877
FP rate for class d = 0.010



J48 -C 0.25 -M 2

○ Use training set
○ Supplied test set Set...
● Cross-validation Folds 10
○ Percentage split % 66

(Nom) class

Result list (right-click for options)
19:07:01 - bayes.NaiveBayes
19:07:08 - trees.J48
19:12:01 - trees.RandomForest

```
Number of Leaves  :     131

Size of the tree :      182


Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1596               92.3611 %
Incorrectly Classified Instances       132                7.6389 %
Kappa statistic                          0.8343
Mean absolute error                      0.0421
Root mean squared error                  0.1718
Relative absolute error                 18.3833 %
Root relative squared error             50.8176 %
Total Number of Instances             1728
```

Status: OK

Fig 2: WEKA J-48 classification.

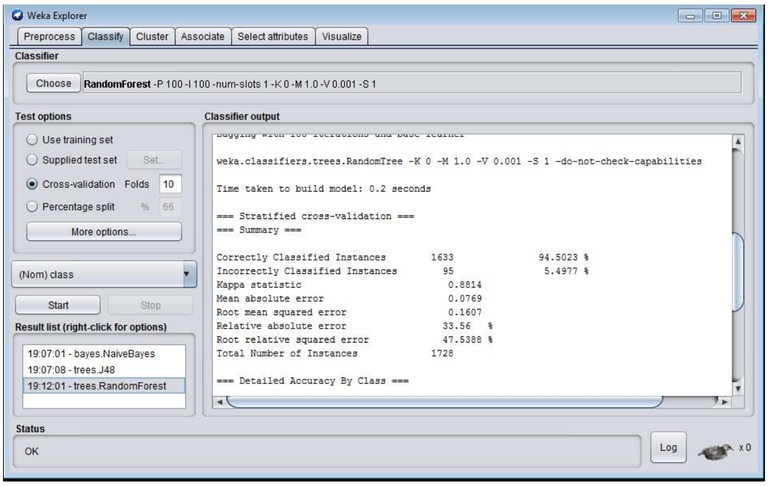
Average TP rate = 0.924
Average FP rate = 0.056
Precision for class a = 0.972
Precision for class b = 0.841
Precision for class c = 0.689
Precision for class d = 0.770
F-measure for class a = 0.962
F-measure for class b = 0.854
F-measure for class c = 0.646
F-measure for class b = 0.820

Table 2: J-48 classifier confusion matrix

N=1728	Predicted:YES	Predicted:NO
Actual:YES	1164 (True positive)	46 (False Negative)
Actual :NO	33 (False Positive)	485 (Negative for unacc)

4.3. Results using Random Forests

Here Class attribute has been chosen from car data set. Random Forests is applied on the data set and the Fig-3 results are being achieved.



RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

○ Use training set
○ Supplied test set Set...
● Cross-validation Folds 10
○ Percentage split % 66

(Nom) class

Result list (right-click for options)
19:07:01 - bayes.NaiveBayes
19:07:08 - trees.J48
19:12:01 - trees.RandomForest

```
Bagging with 100 iterations and base learner

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.2 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1633               94.5023 %
Incorrectly Classified Instances        95                5.4977 %
Kappa statistic                          0.8814
Mean absolute error                      0.0769
Root mean squared error                  0.1607
Relative absolute error                 33.56   %
Root relative squared error             47.5388 %
Total Number of Instances             1728

=== Detailed Accuracy By Class ===
```

Status: OK

Fig 3: WEKA Random Forests classification.

Hence we calculate from Table-3 confusion matrix are:
TP rate for class a = 1173 / (1173+37) = 0.969
FP rate for class a = 12 / (12+506) = 0.023
TP rate for class b = 0.938
FP rate for class b = 0.040
TP rate for class c = 0.580
FP rate for class c = 0.008
TP rate for class d = 0.923
FP rate for class d = 0.010
Average TP rate = 0.945

Table 3: Random Forests classifier confusion matrix

N=1728	Predicted:YES	Predicted:NO
Actual:YES	1173 (True positive)	37 (False Negative)
Actual:NO	12 (False Positive)	506 (Negative for unacc)

Average FP rate = 0.026
Precision for class a = 0.990
Precision for class b = 0.870
Precision for class c = 0.755
Precision for class d = 0.789
F-measure for class a = 0.980
F-measure for class b = 0.902
F-measure for class c = 0.656
F-measure for class b = 0.851

Table 4: classification and cost analysis

Classification Accuracy				Cost analysis		
Class	Naïve-Bayes	J48	Random forest	Naïve-Bayes	J48	Random forest
Unacc	92.88%	95.89%	97.56%	1210	1210	1210
Acc	88.83%	93.75%	94.96%	384	384	384
Good	96.75%	97.33%	98.03%	69	69	69
Vgood	99.30%	98.90%	99.30%	65	65	65

In table-4 the three algorithms were considered for the classification accuracy and cost analysis for the attribute class is shown. The dataset used for this experiment is car.arff dataset. It is an UCI repository dataset which consists of 7 attributes and 1728 instances. The classification Test mode used here is 10-fold cross-validation.

4. Conclusion

The three data mining classification algorithms Naïve Bayes, J48 and Random Forests are used to classify the given sample car data set. Classification accuracy and confusion matrix was calculated for all the three algorithms. Experimental results shows that the Random Forests classification algorithms yields better result by achieving accuracy of 94.05%. F-measure were also calculated to check the accuracy of all three classification algorithms. Cast analysis is same for all the algorithms. In our experiment we use the uci machine learning repository car.arff data set which is derived from simple hierarchical decision model. This dataset is very much useful for testing constructive induction and structure discovery methods using various data mining algorithms. In future the proposed method can be extended to various autonomous data sets.

5. References

1. Sailesh Conjeti, Amin Katouzian. Supervised domain adaptation of decision forests: Transfer Of models trained *in vitro* for *in vivo* intravascular ultrasound tissue characterization, Elsiver, Medical Image Analysis. 2016; 32(s):1-17.
2. Kancherla Jonah Nishanth, Vadlamani Ravi. Probabilistic Neural Network based Categorical Data Imputation, Neurocomputing. http://dx.doi.org/10.1016/j.neucom 2016.08.044.
3. Sushilkuma Rameshpant R, Kalmegh. Comparative Analysis of WEKA Data Mining Algorithm Random Forest, Random Tree and LAD Tree for Classification of Indigenous News Data, International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, 2015; 5:1.
4. Haykin S. Neural Networks: A Comprehensive Foundation, 2nd Ed, Prentice Hall 1999.
5. Aman Kumar Sharma, Suruchi Sahni. A Comparative Study of Classification Algorithms for Spam Email Data Analysis, IJCSE. 2011; 3(5):1890-1895.
6. Kaushik Raviya H, Biren Gajjar. Performance Evaluation of different data mining classification algorithm using WEKA ISSN-2250-1991, 2013; 2:1.
7. Pawar S, Gawanda H. Text Categorization, International Journal of Machine Learning and Computing. 2012; 2:4.
8. Anshul Goyal, Rajni Mehta. Performance Comparison of Naïve Bayes and J48 Classification Algorithms International Journal of Applied Engineering Research. 2012; 7:11. ISSN 0973-4562.
9. Tina Patil R, Mrs Sherekar SS. Performance Analysis of Naive Bayes and J48 Classification Algorithm for Data Classification International Journal of Computer Science and Applications. 2013; 6(2). ISSN:0974-1011.
10. Russell Greiner and Jonathan Schaffer, Exploratorium – Decision Trees, Canada. 2001. URL: http://www.cs.ualberta.ca/~aixplore/ learning/ Decision Trees.
11. Hong Bo Li, Wei Wang, Hong Wei Ding, Jin Dong. Trees Weighting Random Forest Method for Classifying High-Dimensional Noisy Data IEEE 7th International Conference, Publication Year, 2010, 160-163.
12. Margaret Danham H, Sridhar S. Data mining, Introductory and Advanced Topics, Person education, 1st ed, 2006.