NAIVE BAYES CLASSIFIER, DECISION TREE AND ADABOOST ENSEMBLE ALGORITHM – ADVANTAGES AND DISADVANTAGES

Neli Kalcheva[1]
Maya Todorova[2]
Ginka Marinova[3]

DOI: https://doi.org/10.31410/ERAZ.2020.153

Abstract: *The purpose of the publication is to analyse popular classification algorithms in machine learning. The following classifiers were studied: Naive Bayes Classifier, Decision Tree and AdaBoost Ensemble Algorithm. Their advantages and disadvantages are discussed. Research shows that there is no comprehensive universal method or algorithm for classification in machine learning. Each method or algorithm works well depending on the specifics of the task and the data used.*

Keywords: *Classification, Machine learning, Naive Bayes classifier, Decision tree, Ada Boost Ensemble algorithm.*

1. INTRODUCTION

In machine learning, classification is defined as a task for determining the class of an unknown object on the basis of empirical data. There is a relationship between an object and a class that is unknown. The analysis of the nature and characteristics of the objects are directly related to the synthesis of the classification model. This is the main reason why no universal and unambiguous method for classification of objects has been found.

In the exposition we look at the following algorithms:

- probabilistic algorithm – Naive Bayes classifier,
- logical - Decision trees,
- ensemble algorithm - AdaBoost.

2. CLASSIFIERS

2.1. Naive Bayes classifier

One of the classical algorithms in machine learning is the Naive Bayesian Classifier, which is based on Bayes' theorem for determining a posteriori probability of an event.

Bayes' theorem

$$P(y = c|x) = \frac{P(x|y = c)P(y = c)}{P(x)} \quad (1)$$

where:

[1] Technical University of Varna, Bulgaria, 9010 Varna, str. Studentska 1
[2] Technical University of Varna, Bulgaria, 9010 Varna, str. Studentska 1
[3] Technical University of Varna, Bulgaria, 9010 Varna, str. Studentska 1

P(y=c|x) is the probability that an object belongs to class c (a posteriori probability of the class)
P(x|y=c) – probability that the object x is in the range of the class c
P(y=c) – unconditional probability of encountering an object y in class c (a priori probability of the class)
P(x) – unconditional probability of the object x

The purpose of the classification is to determine to which class the object x belongs. Therefore, it is necessary to find the probability class of the object x, i.e. it is necessary to choose from all classes the one that gives the maximum probability P (y = c | x)

$$c_{opt} = \arg\max_{c \in C} P(x|y = c)P(y = c) \quad (2)$$

Known algorithms of the type Naive Bayes classifier are: Bernoulli, Multinomial and Gaussian, associated with different assumptions for the distribution of features.

Advantages:
- main strength is its efficiency; training and classification can be accomplished with one pass over the data (Manning, 2008);
- robust to noise features (Manning, 2008);
- requires a small amount of training data to estimate the parameters necessary for classification. (Khan, 2010, pp. 4-20, Raghunath, 2019, pp. 149-153);
- Works with a small amount of data, handles multiple classes (Harrington, 2012)

Disadvantages:
- conditional independence assumption can be violated by real-world data (Khan, 2010, pp. 4-20, Manning, 2008);
- perform very poorly when features are highly correlated (Khan, 2010, pp. 4-20);
- does not consider frequency of word occurrences. (Khan, 2010, pp. 4-20);
- sensitive to how input data is prepared (Harrington, 2012);

2.2. Decision trees

The Decision tree algorithm (Classification trees) is a powerful tool for presenting and solving classification problems and it is an example of a logical classification methods. The model is built on the basis of statistical estimates and a series of checks on logical conditions.

The tree as a structure in graph theory is described as an oriented graph without cycles. Each tree structure consists of nodes and connections (arcs) between them, each node having one input and two or more output arcs to adjacent nodes.

The root of the tree corresponds to the initial state of the classification process. Each internal node is connected to a logic function that calculates the branch condition. Usually the division is based on one of the characteristics of the input quantity or on a predefined set of division rules. The initial state corresponds to the first logical check to be performed. Each internal, intermediate node corresponds to the following logical checks. The end nodes, the leaves, represent the resulting classes. When traversing the tree, starting from the root, each node divides the set of solutions until it reaches the final node that defines the class. Each inner node has an associated dividing predicate – i.e. logical function related to the calculation of the branching condition.

The classification is considered complete when one of the leaves is reached (end nodes, nodes without heirs). The value of this leaf determines the class to which the object in question belongs. The classification tree model can be represented as a series of control structures of the if-then-else type.

The main parameters of the decision tree algorithm concern the different ways of choosing division rules and rules for ending the growth of the tree.

For each node of the tree, an optimal splitting rule is selected, including the selection of an attribute and the value of that attribute by which to split. The selection of an attribute is based on the information gain that comes from dividing the objects by the respective attribute.

The information gain of the A attribute, in regards to a set of examples S, is calculated as such:

$$Gain(S,A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} E(S_v) \quad (3)$$

where Sv e set of examples, which have a value of v for the attribute A, E(S) - entropy of the S set of examples.

Entropy is defined as such:

$$E(S) = \sum_{i=1}^{n} -p_i \log_2(p_i) \quad (4)$$

where

n – number of examples

p_i – the probability of picking i class.

Some algorithms that implement decision trees are the following CART (Classification And Regression Tree), ID3 (Iterative Dichotomiser 3), C4.5 (successor of ID3) and its subsequent version C5.0, CHAID (Chi-square automatic interaction detection), CN2, NewId, ITrule and others.

Advantages:

- intuitively understandable (Gareth, 2015);
- visually clear (Gareth, 2015)
- able to train data that contains errors or missing values (Batura, 2017);
- computationally cheap (Harrington, 2012);
- fast and scalable, both in the number of variables, and the size of the training set (Hotho, 2005, pp.19-62);

Disadvantages:

- prone to overfitting (Harrington, 2012);
- the final decision can depend on relatively few terms, when dealing with text mining (Hotho, 2005, pp.19-62);
- needs a large volume of data for accurate results (Batura, 2017);
- can lead to large and excessively complex tree structures, when the dataset has a large number of entries (Khan, 2010, pp. 4-20).

3. ADABOOST

The adaptive boosting algorithm AdaBoost is an example of an ensemble boosting algorithm. This is an iterative algorithm in which the „strong" classifier makes small learning errors based on the „weak" classifier, which correctly classifies more than 50%. In the ensemble algorithm AdaBoost, the different classifiers are trained sequentially. Each new classifier is trained based on the effectiveness of existing trained classifiers. The reinforcement consists in focusing the new classifiers on data that have been misclassified by previous classifiers.

The AdaBoost adaptive algorithm calculates weight for each example in the training data. A „weak" classifier is trained first. The errors are calculated, after which the „weak" is trained a second time with the same data set. Next, the weights of the training set are adjusted so that the correctly classified examples from the first iteration are weighed less, and the examples that are incorrectly classified in the first iteration are weighed more.

To obtain an answer from all weak classifiers, the AdaBoost algorithm determines the α values of each classifier (6). The α values are based on the error of each weak classifier.

The error ε is calculated using the following formula:

$$\varepsilon = \frac{count\ of\ incorrectly\ classified\ examples}{total\ number\ of\ examples} \tag{5}$$

α is calculated through ε

$$\alpha = \frac{1}{2}\ln\left(\frac{1-\varepsilon}{\varepsilon}\right) \tag{6}$$

Advantages:

- low generalization error; easy to code (Harrington, 2012);
- computationally efficient (Witkin, 2011);
- applicable to complex tasks (Witkin, 2011);
- able to be easily modified (Witkin, 2011);
- flexible; can be easily combined with other learning algorithms (Jaakkola, 2012);

Disadvantages:

- sensitive to outliers (Harrington, 2012);
- there can be a lot of noise when training (Voronov, 2008);
- needs a large sample (Voronov, 2008);
- can lead to "unwieldy" compositions (Voronov, 2008).

4. CONCLUSION

Research shows that there is no comprehensive universal method or algorithm for classification in machine learning, with a 100% accuracy for any applied task. Each method or algorithm works well depending on the specifics of the task and the data used. Thus, many different algorithms ought to be explored before picking the right one. The strengths and weaknesses of each one should be noted when making a decision.

REFERENCES

Gareth, J., D. Witten, T. Hastie, R. Tibshirani, *An Introduction to Statistical Learning with Applications in R*, ISBN 978-1-4614-7138-7, pp.1-pp.440, 2015

Harrington, P. *Machine Learning in Action*. Greenwich, CT, USA: Manning, 2012.

Hotho, A., A. Nürnberger, G. Paab. A brief survey of text mining. *LDV Forum-GLDV J. Comput. Linguistics Lang. Technol*., vol. 20, no. 1, 2005, pp. 19-62.

Jaakkola, T. MIT CSAIL, The AdaBoost algorithm, people.csail.mit.edu/dsontag/courses/ml12/slides/lecture13.pdf

Khan, A., B. Baharudin, L. Lee, K. Khan. A review of machine learning algorithms for text-documents classification. *Journal of Advances Information Technology*, vol. 1, 2010, pp. 4-20.

Manning, C., P. Raghavan, H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

Raghunath, D., C. Usha, K. Veera, V. Manoj. Predicting heart disease using machine learning techniques. *International Research Journal of Computer Science* VI, no.4, 2019, pp. 149-153.

Rokach, L., O. Maimon, *Decision Trees, Data Mining and Knowledge Discovery Handbook*, ISBN978-0-387-25465-4, 2005

Batura, T. Metody avtomaticheskoj klassifikacii tekstov. Programmnye produkty i sistemy 30, no. 1, 2017.

Voronov, A. Obzor algoritmov mashinnogo obuchenija, Kurs lekcij K.V. Voroncova po mashinnomu obuchenij (2007-2008), https://www.compression.ru/video/seminar/slides/2009_MachineLearning.pdf.

Uitkin, L. Machine Learning Kompozicionnye metody mashinnogo obuchenija https://tema.spbstu.ru/machine_learning/