IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 2, February 2024

DOI: 10.17148/IARJSET.2024.11220

Study of Multiclass Classification Techniques

Komal Shah[1], Dr. Kajal S. Patel[*2]

Research Scholar, Gujarat Technological University, Ahmedabad, Gujarat, India[1]

Associate Professor, Vishwakarma Government Engineering College affiliated to Gujarat Technological University, Ahmedabad, Gujarat, India[*2]

Abstract: Classification is supervised learning technique in data mining that produces a learned model from a labelled dataset. Many classification algorithms are available to solve binary classification problem where there are only two possible values of class label. Multiclass classification deals with the classification problem where class label has more than two possible values. Multiclass classification problems are highly relevant in recent machine learning developments, because there are numerous real life applications that rely on high number of parameters and classes. For example genetics, banking, physics, medical and other applications. In this paper we discussed techniques of multiclass classification and challenges in field of data mining.

Keywords: Classification, Data Mining, Multiclass Data, One-vs-One, One-vs-All, ECOC

I. INTRODUCTION

Classification involves determining which category, or class, a given instance belongs to within a predefined set of categories. When these categories encompass more than two distinct possible class labels, the task is referred to as multiclass classification. In multiclass classification, it is assumed that each instance is assigned only one class label (in contrast to multi-label classification, where multiple class labels can be assigned to a single instance), and that the class labels are independent, meaning there are no inherent relationships among them. For instance, in a multiclass classification problem, one might classify temperatures into low, moderate, and high categories, or identify types of infections as viral, bacterial, fungal, etc.

Consider C as the set comprising potential class label values, and let D represent a dataset comprising labeled instances in the form (xi, yi), where xi ∈ D is an m-tuple consisting of input attributes, and yi ∈ C is the target attribute, denoting the class. The objective of classification is to discover a learning model, H, such that H(xi) = yi for new, unseen instances. If C = {c1, c2}, the problem is succinctly defined as a two-class or binary classification problem. However, if C = {c1, c2, ..., cn}, where n > 2, then it constitutes a multiclass classification problem. Various algorithms have been proposed to tackle binary classification problems, with some adaptable to the multiclass scenario, while others require specific formulations to address multiclass classification.

Multiclass classification can be performed using these techniques: (i) Extension of binary classifiers for multiclass classification. (ii) Decomposition of the multiclass problem into binary classification problems. Each of these techniques is discussed in detail below.

II. EXTENSION OF BINARY CLASSIFIER FOR MULTICLASS CLASSIFICATION

In this technique multiclass classification problem is solved by applying binary class classifiers. These binary classifiers are like Decision Tress, Neural Networks, k-Nearest Neighbour, Naive Bayes and Support Vector Machines.

2.1. Decision Trees

Decision trees are a robust classification technique utilized in supervised learning. Among the well-known algorithms for constructing decision trees are Classification and Regression Trees [1]. These trees are structured models that build classification or regression models in a hierarchical tree format. The construction of a decision tree involves iteratively dividing a dataset into smaller subsets based on various attributes. A decision tree comprises decision nodes and leaf nodes, with each decision node having two or more branches leading to subsequent nodes or leaf nodes. Leaf nodes represent final classifications or decisions. Decision trees begin with the root node, which is usually the most influential predictor. They exhibit versatility in handling both categorical and numerical data. To classify an unknown instance, one follows a path from the root to a leaf node of the decision tree, where at each node, a test is conducted on an attribute of the dataset. Once a leaf node is reached, its associated class label is assigned, accommodating both binary and multiclass classification problems.

DOI: 10.17148/IARJSET.2024.11220

	Cough	Fever	Weight	Pain	Class Label
Mary	no	yes	normal	throat	Flu
Joy	no	yes	normal	abdomen	Appendicitis
Eva	yes	yes	skinny	none	Flu
Tom	yes	no	obese	chest	heart disease

Table 1: Sample Database

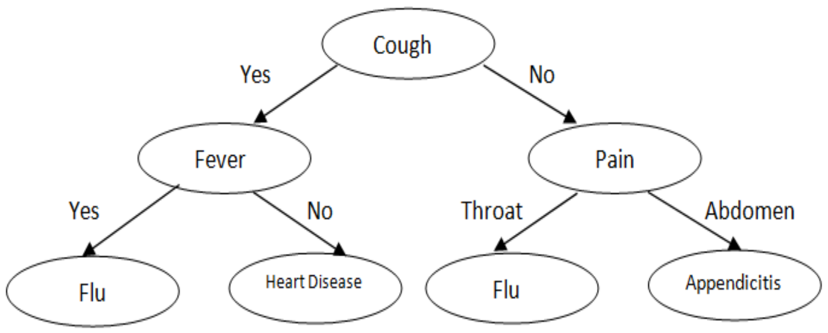


Figure 1: Decision Tree of data in Table 1

Consider sample database of Table 1. Here cough is most predictable attribute and root of decision tree. If a person is suffering from cough and fever then this decision tree predicts flu as class label. In the same way if person is not having cough but abdomen pain this decision tree predicts appendicitis as class label. There are many decision tree algorithms some are ID3, C4.5 [2], CART, etc. Decision tree are computationally simply to understand and interpret. They can handle both numerical and categorical data and performs well on large data in a short time. Sometimes decision trees fail to generalize data by creating over complex trees.

2.2. Neural Networks

Neural network operates within the framework of supervised learning, where a neural network receives input instances along with their corresponding correct class labels. A multilayer feed-forward neural network typically comprises layers of input neurons, one or more layers of hidden neurons, and a layer of output neurons [1], as displayed in Figure 2. Being a "feed-forward" neural network, it does not permit cycles from later layers to earlier ones.

Multilayer feedforward neural networks are adept at addressing both binary and multiclass classification problems. In a binary classification scenario, the output layer consists of a single neuron. However, in multiclass classification, the output layer typically comprises N binary neurons. The determination of the output codeword for each class value can be achieved through either one-per-class coding or distributed output coding [4].

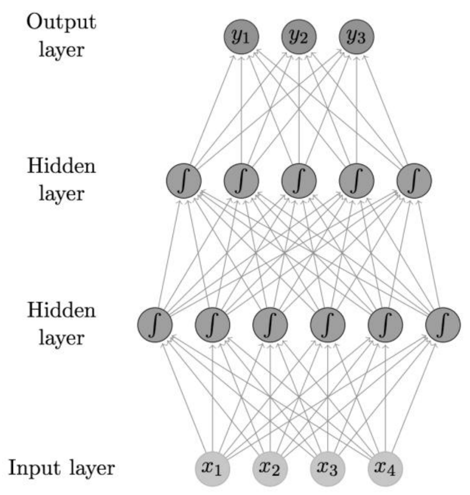


Figure 2: Sample Multi Layer Feed Forward Neural Network

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 11, Issue 2, February 2024

DOI: 10.17148/IARJSET.2024.11220

In the one-per-class coding technique, each output neuron is assigned to identify a specific potential value of the class label. For a given class label value, the code of the corresponding output neuron should be 1, while all other output neurons' codes for that class label value should be 0. Hence, this method requires N neurons in the output layer, where N represents the number of potential values for the class label. When assessing an unknown instance, the output neuron representing the class label for that instance should have the highest activation. For instance, in a three-class problem with class labels {c1, c2, c3}, the output codes are illustrated in Table 2.

Class Label	Codeword
c1	100
c2	010
c3	001

Table 2: One per Class Coding

Class Label	Codeword
c1	00000
c2	00111
c3	11100

Table 3: Distributed Output Coding

In the distributed output coding approach, each possible combination of 1-bits, where N represents the number of class labels, is assigned a unique binary codeword comprising N output neurons. When evaluating an unfamiliar instance, it is compared against existing codewords, and the closest codeword wins, determining the assigned class label. Typically, the Hamming distance is utilized to measure the proximity between codewords, which quantifies the differing bits between them.

For instance, in a three-class scenario utilizing 5-bit codewords, the codes could be depicted as shown in Table 3. If an unknown instance produces a codeword like 11101 upon evaluation, its distance is computed from the codewords listed in Table 3. The two nearest codewords are found, with one corresponding to class c3, having a distance of 1, hence assigning the instance the class value c3.

2.3. k Nearest Neighbour

The k-Nearest Neighbors (kNN) technique is a widely utilized classification method [5]. In kNN classification, the output entails determining the class membership of an instance. This is achieved through a majority vote among its nearest neighbors. Specifically, the most common class label among its k closest neighbors is assigned to the unknown instance. Typically, the value of k is a positive integer and remains relatively small. For instance, if k equals 1, the instance is straightforwardly assigned the class value of its single nearest neighbor. The selection of the optimal value for k is usually determined through a validation set or via cross-validation.

The kNN classification algorithm initially identifies a group of k instances from the training set that are closest to the test instance [6]. To classify an unknown instance, kNN algorithm finds the distance between this instance and the labeled instances, identifies its k-nearest neighbors, and subsequently utilizes the class values of these nearest instances to ascertain the class value of the unknown instance.

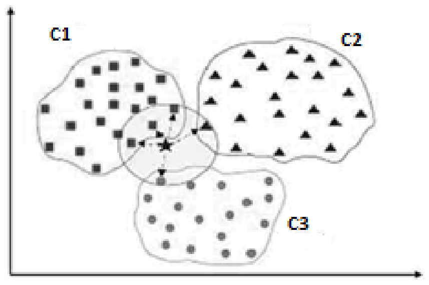


Figure 3: Example of kNN Classification for 3 class

DOI: 10.17148/IARJSET.2024.11220

Consider example given in figure 3. There are classes - c1, c2 and c3. Suppose unknown instance is shown by star in figure 3. kNN finds nearest neighbours to this black instance. For calculating distance between two existing instances, various methods like Euclidean distance, Manhattan distance, the hamming distance or cosine distance are used. Value of k defines number of neighbouring instances to be considered.

Here value of k is 5 and that why 2 instances from c1, 1 instance from c2 and 1 instance from c3 is considered in above figure and hence c1 is assigned to unknown instance. kNN is a lazy learner and requires less time in learning but needs all training data while determining class label of unknown instance.

2.4. Naive Bayes

The Naive Bayes algorithm operates as a probabilistic classification method founded on the principles of Bayes' theorem [7]. It makes strong independence assumptions regarding attributes and posits that the presence or absence of a specific attribute within a class is independent of the presence or absence of any other attribute, given the class variable. This approach proves to be beneficial when handling high-dimensional input data.

Consider a scenario with m classes, denoted as C1, C2, ..., Cm. Given a tuple, X, the classifier predicts that X belongs to the class with the highest value of posterior probability conditioned on X. In essence, the Naive Bayesian classifier assigns tuple X to class Ci if and only if $P(Ci|X) > P(Cj|X)$ for $1 <= j <= m; j \neq i$: The posterior probability for each and every class label is calculated using following equation

$$P(Ci|X) = \frac{P(X|Ci)P(Ci)}{P(X)}$$

2.5. Support Vector Machines

Support Vector Machines (SVM) represent a supervised learner wherein each given instance is plotted in an n - dimensional space [8, 9]. Here, n denotes the number of attributes in the dataset, with each attribute's value serving as the coordinate for a particular dimension. Classification is executed by identifying the hyperplane that differentiates between the two classes, as depicted in Figure 4. Among the three planes illustrated, plane B demonstrates superior classification.

Classification problems can be broadly categorized as linearly separable problem and non-linearly separable problem. Linearly separable problems can be readily segregated by a straight hyperplane, whereas non-linearly separable problems cannot. To address non-linearly separable issues, features are transformed from the initial space to a higher-dimensional feature space, wherein the separation of input data becomes more feasible.

The kernel function [10] is applied to each data instance to map the original non-linear instances into a higher-dimensional space, where they can be effectively separated.

While the basic SVM directly works only for binary class classification task, various extensions [11, 12, 13, 14, 15] have been designed to accommodate multiclass classification as well. These extensions introduce additional constraints to effectively handle the differentiation of different class values.

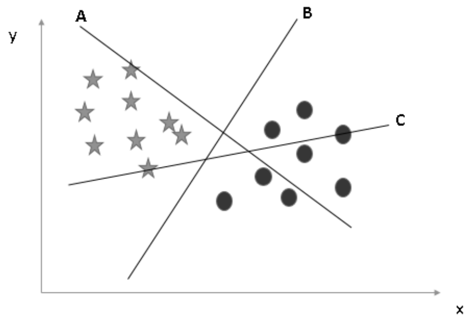


Figure 4: Support Vector Machine

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁘ Peer-reviewed & Refereed journal ⁘ Vol. 11, Issue 2, February 2024

DOI: 10.17148/IARJSET.2024.11220

III. DECOMPOSING MULTICLASS PROBLEM INTO BINARY CLASS CLASSIFICATION PROBLEM

In this technique multiclass classification problem is divided into several binary classification problems that can be solved using traditional binary classifiers. Several techniques are proposed for decomposition [16, 17, 18]. They are categorized as (i) one-vs-all (ii) one-vs-one (iii) Error correcting output coding. These techniques are discussed in following sections.

3.1. One-vs-All (OVA)

In this approach, the task of classifying among N classes is simplified by breaking it down into N binary classification problems [19]. Each binary classifier is responsible for distinguishing a specific class from the remaining N−1 classes. Thus, N binary classifiers are required for this technique. During training, each classifier is trained with positive instances belonging to its respective class and negative instances from the other N−1 classes.

When presented with an unknown instance for testing, the classifier that yields the highest output is deemed the winner, and its associated class value is assigned to the instance. The simplicity of this technique, as outlined by the authors in [19], offers comparable performance to more complex methods, especially when the binary classifiers are well-tuned. However, it's important to note that this technique demands a higher memory overhead, approximately proportional to the square of the total number of training instances.

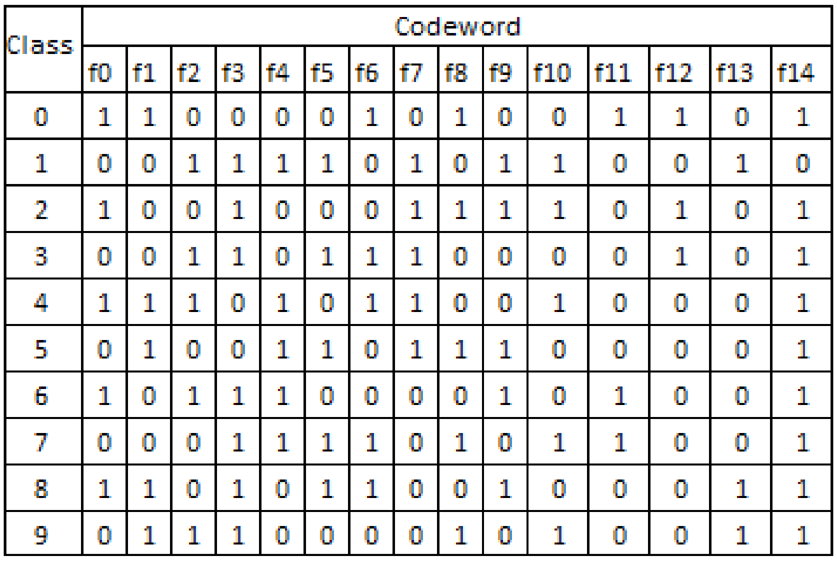
3.2. One-vs-One (OVO)

In this approach, every class is systematically compared with each other [20, 21]. It involves constructing a binary classifier for distinguishing between each pair of classes, while disregarding the remaining classes. If there are a total of N classes, this method necessitates the creation of N*(N−1)/2 binary classifiers.

During the testing phase, classification is conducted by a voting mechanism among these classifiers, with the class garnering the highest number of votes being assigned. While this technique generates a greater number of classifiers compared to the One-vs-All approach, the training instances for each classifier are relatively small. Research results [16, 18] indicate that this method yields superior accuracy compared to the One-vs-All technique.

3.3. Error correcting output coding (ECOC)

It is an ensemble technique designed for multi class classification problem. This technique uses idea of code words similar to the code words used in neural networks [17]. This technique converts N class classification problem into a large number of binary classification problems. ECOC technique assigns a unique code word to each class and instead of using class label for classification it uses this unique code.



Class	Codeword														
	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14
0	1	1	0	0	0	0	1	0	1	0	0	1	1	0	1
1	0	0	1	1	1	1	0	1	0	1	1	0	0	1	0
2	1	0	0	1	0	0	0	1	1	1	1	0	1	0	1
3	0	0	1	1	0	1	1	1	0	0	0	0	1	0	1
4	1	1	1	0	1	0	1	1	0	0	1	0	0	0	1
5	0	1	0	0	1	1	0	1	1	1	0	0	0	0	1
6	1	0	1	1	1	0	0	0	0	1	0	1	0	0	1
7	0	0	0	1	1	1	1	0	1	0	1	1	0	0	1
8	1	1	0	1	0	1	1	0	0	1	0	0	0	1	1
9	0	1	1	1	0	0	0	0	1	0	1	0	0	1	1

Table 5: ECOC Example

Table 5 shows a 15 bit ECOC for 10 class problem. Each class is assigned a unique binary code of length 15. For example, class 3 has the codeword 001101110000101. During training phase, one binary classifier is learned for each column.

For example, for third column, it builds a binary classifier to separate {0, 2, 5, 7, 8} from {1, 3, 4, 6, 9}. Thus 15 binary classifiers are trained in this way. To classify unknown instance, all 15 binary classifiers are evaluated to obtain a 15 bit code word. Finally, this technique chooses the class label whose codeword is closet to output code word as the predicted label.

IV. COMPARING DECOMPOSITION VS EXTENSION FOR MULTICLASS CLASSIFICATION

Comparing decomposition in binary classification problems versus applying existing classifiers for multi-class classification involves evaluating their effectiveness, computational efficiency, and ease of implementation.

Decomposition techniques typically divide the multi-class problem into multiple binary classification tasks. Each binary classifier focuses on distinguishing between one class and the rest (One-vs-All or One-vs-One). This approach can simplify the problem and may lead to better performance, especially when dealing with complex class distributions or imbalanced datasets. On the other hand, applying existing classifiers directly to multi-class problems may not fully exploit the inherent structure of the data, potentially resulting in suboptimal performance.

Decomposition techniques can increase computational complexity, as they involve training and evaluating multiple binary classifiers. However, the computational overhead may be manageable depending on the size of the dataset and the complexity of the classifiers used. Applying existing classifiers directly to multi-class problems may be computationally more efficient, especially if the classifiers have built-in support for handling multiple classes efficiently.

Decomposition techniques require additional preprocessing steps to divide the multi-class problem into binary tasks and combine the results for final prediction. This may involve careful selection of binary classifiers and strategies for combining their outputs. In contrast, applying existing classifiers directly to multi-class problems can be simpler to implement, especially if the classifiers have native support for multi-class classification.

V. CONCLUSION

This paper investigates various techniques for addressing multi-class classification problems. The first technique involves extending and directly applying binary classification methods to handle multi-class classification. This encompasses employing decision trees, neural networks, support vector machines, naive Bayes, and k-nearest neighbors.

The second technique entails breaking down the multi-class problem into several binary classification problems. Various methods such as One-vs-All (OVA), One-vs-One (OVO), and Error Correcting Output Codes (ECOC) are commonly utilized for this purpose.

In summary, the choice between decomposition and applying existing classifiers depends on factors such as the nature of the dataset, the computational resources available, and the specific requirements of the application. Decomposition techniques may offer better performance but require additional computational resources and implementation effort, while applying existing classifiers directly may be more computationally efficient and easier to implement but may not fully exploit the structure of the data.

REFERENCES

[1]. L. Breiman, J. Friedman, R. A. Olshen, and C. J. Stone. Classification and Regression Trees. Chapman and Hall, 1984.
[2]. J. R. Qufinlan. C4.5: Programs for Mahine Learning. Morgan Kaufmann, 1993.
[3]. Daniel Svozil, Vladimir KvasniEka, JiE Pospichal, Introduction to multi-layer feed-forward neural networks, Chemometrics and Intelligent Laboratory Systems 39 (1997) 43-62.
[4]. Christopher M. Bishop, editor. Neural Networks for Pattern Recognition. Oxford University Press, 1995.
[5]. Stephen D. Bay. Combining nearest neighbor classifiers through multiple feature subsets. In Proceedings of the 17th International Conference on Machine Learning, pages 37–45, Madison, WI, 1998.

[6]. Yun-lei Cai, Duo Ji ,Dong-feng Cai,A KNN Research Paper Classification Method Based on Shared Nearest Neighbor, Natural Language Processing Research Laboratory, Shenyang Institute of Aeronautical Engineering, Shenyang, China, June 15–18, 2010
[7]. Irina Rish. An empirical study of the naive bayes classifier. In IJCAI Workshop on Empirical Methods in Artificial Intelligence, 2001.
[8]. Corinna Cortes and Vladimir Vapnik. Support-vector networks. Machine Learning, pages 273–297, 1995.
[9]. C. J. Burges. A tutorial on support vector machines for pattern recognition. In Data Mining and Knowledge Discovery, pages 1–47, 1998.
[10]. Tomer Hertz Tomboy, Aharon Bar Hillel and Aharonbh Daphna Weinshall, "Learning a Kernel Function for Classification with Small Training Samples," School of Computer Science and Engineering, The Center for Neural Computation, The Hebrew University of Jerusalem, Jerusalem, Israel.
[11]. J. Weston and C. Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway, University of London, 1998.
[12]. Erin J. Bredensteiner and Kristin P. Bennett. Multicategory classification by support vector machines. Computational Optimization and Applications, 12:53–79, January 1999.
[13]. Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. Journal of Machine Learning Research, pages 265–292, 2001.
[14]. Yoonkyung Lee, Yi Lin, and Grace Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. Journal of the American Statistical Association, 99(465):67–81, March 2004.
[15]. Chen, S. G., and Juan Xu. "Least squares twin support vector machine for multi class classification." *International Journal of Database Theory & Application* 8.5,pages 65-76, 2015
[16]. Erin Allwein, Robert Shapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. Journal of Machine Learning Research, pages 113–141, 2000.
[17]. T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error correcting output codes. Journal of Artificial Intelligence Research, 39:1–38, 1995.
[18]. Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. In IEEE TRANSACTIONS ON NEURAL NETWORKS, volume 13, pages 415–425, March 2002.
[19]. Ryan Rifkin and Aldebaro Klautau. Parallel networks that learn to pronounce english text. Journal of Machine Learning Research, pages 101–141, 2004.
[20]. J. Friedman. Another approach to polychotomous classification. Technical report, Stanford University, 1996.
[21]. Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. In Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors, Advances in Neural Information Processing Systems, volume 10. The MIT Press, 1998.