Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 06, June-2021

Message Text Classifier

P. Malathi Asst. Prof, Dept of ISE
Dr. Ambedkar Institute of Technology,
Outer Ring Road, Near Jnanabharthi Campus, Mallathahalli,
Bengaluru- 560056

***Abstract*— In this work, classification of messages using Weka ID3 classifier is proposed. We use feature testing to generate ARFF files which act as input for the Weka ID3 to build the decision tree. The decision tree thus generated is used to predict the results. The results emerged from have quick response time in classifying the messages. Message Classifier, which aims to assign a Short Message Service (SMS) message to two categories based on its contents, is a fundamental task for building that allow individuals to construct classifiers that have relevance for a variety of domains.**

Keywords- Artificial Intelligence, machine learning, Iterative Dichotomizer (ID3), Decision tree, entropy, splitting attribute,information gain

I. INTRODUCTION

As the mobile phone market is rapidly expanding and the modern life is heavily dependent on cell phones, Short Message Service (SMS) has become one of the important media of communications. This media of communication has been considered as one of the fundamental and primitive way of connection for its cheapness, more convenient for advanced to novice users of cell phone, mobility, individualization and documentation.

SMS classifying technology has important significance to assist people in dealing with SMS messages. Although SMS classification can be performed with little or no effort by people, it still remains difficult for computers. Machine learning offers a promising approach to the design of algorithms for training computer programs to efficiently and accurately classify short text message data.

II. PREVIOUS WORK

A. Text Classification is the process of classifying documents into predefined classes based on its content. Text classification is important in many web applications like document indexing, document organization, spam filtering etc. [2]. In text classification, a text messages may partially match many categories. We need to find the best matching category for the text messages.

A good text classifier is a classifier that efficiently categorizes large sets of text documents in a reasonable time frame and with an acceptable accuracy, and that provides classification rules that are human readable for possible fine-tuning. If the training of the classifier is also quick, this could become in some application domains a good asset for the classifier. Many techniques and algorithms for automatic text categorization have been devised.

Classification is an important task in both data mining and machine learning communities, however, most of the learning approaches in text categorization are coming from machine learning research. A number of text classification techniques have been applied including, Naive Bayes [2,3] k-NN[4], Neural Network [5], centroid-based approaches [9,10], Decision Tree [12], SVM [13], Rocchio Classifier [7], Regression Models [11], Bayesian probabilistic approaches, inductive rule learning, and Online learning [14,15].

B. Limitations of Existing systems

Naive-Bayes theorem can't learn interactions between features. Term Frequency-Inverse Document Frequency(TF-IDF) has several limitations. It computes document similarity directly in the word-count space, which may be slow for large vocabularies. It assumes that the counts of different words provide independent evidence of similarity. It makes no use of semantic similarities between words. Neural networks have been criticized for their poor interpretability. SVM is a binary classifier. To do a multi-class classification, pair-wise classifications can be used (one class against all others, for all classes). Computationally expensive, thus runs slow.

III. ALGORITHM

Very simply, ID3 builds a decision tree from a fixed set of examples. The resulting tree is used to classify future samples. The example has several attributes and belongs to a class (like yes or no). The leaf nodes of the decision tree contain the class name whereas a non-leaf node is a decision node. The decision node is an attribute test with each branch (to another decision tree) being a possible value of the attribute. ID3 uses information gain to help it decide which attribute goes into a decision node. The advantage of learning a decision tree is that a program, rather than a knowledge engineer, elicits knowledge from an expert.

ID3 is a non incremental algorithm, meaning it derives its classes from a fixed set of training instances. An incremental algorithm revises the current concept definition, if necessary, with a new sample. The classes created by ID3 are inductive, that is, given a small set of training instances, the specific classes created by ID3 are expected to work for all future instances. The distribution of the unknowns must be the same as the test cases. Induction classes cannot be proven to work in every case since they may classify an infinite number of instances. ID3 uses information gain as its attribute selection measure. Let node N represent or hold the tuples of partition D. The attribute with the highest information gain is chosen as the splitting attribute for node N. This attribute minimizes the information needed to classify the tuples in the resulting partitions and reflects the least randomness or "impurity" in these partitions. The expected information needed to classify a tuple in D is given by

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 06, June-2021

$$\text{Info}(D) = -\sum_{i=1}^{m} p_i \log_2 p_i$$

Info(D) is also known as the entropy of D. Now, suppose the tuples in D are partitioned on some attribute A having v distinct values. These partitions would correspond to the branches grown from node N. The information needed (after the partitioning) to arrive at an exact classification is measured by

$$\text{Info}_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \text{Info}(D_j)$$

$\text{Info}_A(D)$ is the expected information required to classify a tuple from D based on the partitioning by A. The smaller the expected information (still) required, the greater the purity of the partitions. Information gain is defined as the difference between the original information requirement (i.e., based on just the proportion of classes) and the new requirement (i.e., obtained after partitioning on A).

In other words, Gain(A) tells us how much would be gained by branching on A. It is the expected reduction in the information requirement caused by knowing the value of A. The attribute A with the highest information gain, Gain(A), is chosen as the splitting attribute at node N. This is equivalent to saying that it is intended to partition on the attribute A that would do the "best classification," so that the amount of information still required to finish classifying the tuples is minimal (i.e., minimum $\text{Info}_A(D)$).

How does ID3 decide which attribute is the best? A statistical property, called information gain, is used. Gain measures how well a given attribute separates training examples into targeted classes. The one with the highest information (information being the most useful for classification) is selected. In order to define gain, we first borrow an idea from information theory called entropy. Entropy measures the amount of information in an attribute.

Given a collection S of c outcomes

Entropy(S) = S -p(I) $\log_2$ p(I)

where p(I) is the proportion of S belonging to class I. S is over c.

Note that S is not an attribute but the entire sample set.

Algorithm: Generate decision tree. Generate a decision tree from the training tuples of data partition, D.

Input:Data partition, D, which is a set of training tuples and their associated class labels; attribute list, the set of candidate attributes;

Attribute selection method, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a splitting attribute and, possibly, either a split-point or splitting subset.

Output: A decision tree.

Method:

(1) create a node N;

(2) if tuples in D are all of the same class, C, then

(3) return N as a leaf node labelled with the class C;

(4) if attribute list is empty then

(5) return N as a leaf node labelled with the majority class in D; // majority voting

(6) apply Attribute selection method(D, attribute list) to find the "best" splitting criterion;

(7) label node N with splitting criterion;

(8) if splitting attribute is discrete-valued and

Multi way splits allowed then // not restricted to binary trees

(9) attribute list ← attribute list - splitting attribute; // remove splitting attribute

(10) for each outcome j of splitting criterion

// partition the tuples and grow sub trees for each partition

(11) let D_j be the set of data tuples in D satisfying outcome j; // a partition

(12) if D_j is empty then

(13) attach a leaf labelled with the majority class in D to node N;

(14) else attach the node returned by Generate decision tree(D_j, attribute list) to node N;

endfor

(15) return N;

IV. IMPLEMENTATION

1. Data Set Collection and Preprocessing.

A dataset is a collection of data to be used for classification. In this work dataset consists of two different domain *collections*: Financial and Non-Financial. Data pre-processing is an important step in the data mining process. If there is much irrelevant and redundant information present or noisy and unreliable data, then knowledge discovery during the training phase is more difficult. Data preparation and filtering steps can take considerable amount of processing time. Data pre-processing includes cleaning, normalization, transformation, feature extraction and selection, etc. The product of data pre-processing is the final training set and testing set. Here we have collected the messages and stored in excel sheet. We have divided the complete data set into train set and test set and tagged both the sets as '0' or '1' indicating non-financial and financial as part of the preprocessing.

2. Feature Selection.

In machine learning and statistics, feature selection, also known as variable selection, attribute selection or variable subset selection, is the process of selecting a subset of relevant features (variables, predictors) for use in model construction. Feature selection is itself useful, but it mostly acts as a filter, muting out features that aren't useful in addition to your existing features. The objective of variable selection is three-fold: improving the prediction performance of the predictors, providing faster and more cost-effective predictors, and providing a better understanding of the underlying process that generated the data.

There are three general classes of feature selection algorithms- Filter methods, Wrapper methods & Embedded methods.

The table in fig 4.1 contains the examples of features selected for our classification.

Variable	Description
Credited	Key word in financial messages
Currency Indicator	Rs , INR , $ etc
Account Number	3756XXXX125 , **********768, etc
Debited	Keyword in financial messages
Free	Keyword in ads messages
Credit Card No.	5768xxxxx1894, etc
Bonus	Keyword in ads keyword

Fig 4.1: Table with examples of selected features

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 06, June-2021

3. Feature Testing.

This step is very important for creating ARFF files. We check for the existence of each feature for each message in the data set collected. The example for checking if the feature credited exists in the message or not. If it exists then it will return 'yes' else 'no'. Feature testing is mainly used to prediction in the test ARFF.

4. Attribute-Relation File Format (ARFF).

An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of instances sharing a set of attributes. ARFF files have two distinct sections- header followed by data. Sparse ARFF files are very similar to ARFF files, but data with value 0 are not be explicitly represented.

5. Generating Train and Test ARFF files.

Steps to generate Train ARFF - Add the header into ARFF File, read the excel sheet where Training Set is stored, perform Feature Test, write the results of feature test into ARFF file.

Steps to generate Test ARFF - Add the header into ARFF File, read the excel sheet where Test Set is stored, perform Feature Test, write the results of feature test into ARFF file

6. Message Classification

Message classification process is shown in the Fig 4.2 below. The decision tree thus obtained for the financial example taken is shown in the Fig 4.3

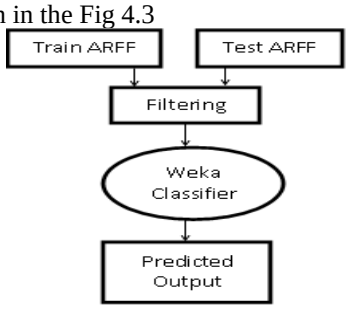

Fig 4.2 Message Classifier

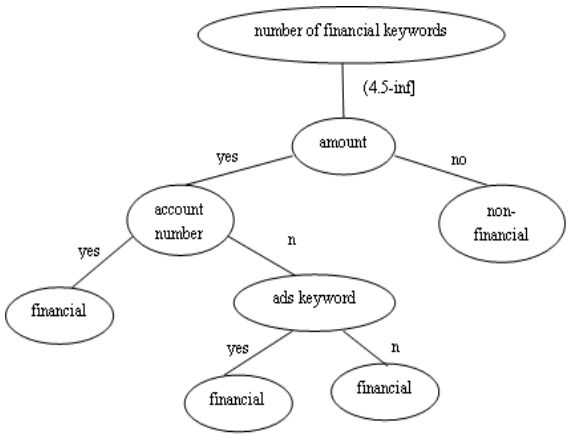

Fig 4.3 Decision Tree of Message Classifier

V. CONCLUSION & FUTURE WORK

The proposed system offers another approach to classification of Short Message Service(SMS) messages in addition to the existing systems. It provides an accurate, efficient method with real time analysis support. The proposed system creates understandable prediction rules from the training data. It also builds the tree fast and short. It only needs to test enough attributes until all data is classified. Finding leaf nodes enables test data to be pruned, reducing number of tests and whole dataset is searched to create tree.

Based on our findings the proposed system is capable of classifying an incoming message with great accuracy that can be extended for future use.The system uses tool with Free availability and Portability. Since it is implemented in the Java programming language it runs on almost any modern computing platforms like Windows, Mac OS X and Linux. The results are also confirmed using weka GUI for accurate results. The features used for training is easy to understand and modifiable. Many classification algorithms limit the broad area of domains for validating text data. In proposed system, a classifier extends the number of domains to text data. At its simplest, system provides a quick and easy way to explore and analyze data. In addition the response time for classifying messages gets drastically reduced.

To be enhanced and implemented in real time in further classifying the financial messages into categories such as income, expense, informational and so on. To be enhanced and implemented in real time in estimating the expenses and current financial status of an individual over a specific period of time using the Short Message Service (SMS) messages the individual has received. To extend this classification to electronic-mail and online messaging services.

REFERENCES

[1] A.Selamat, 2003. Studies on Mobile Agents for Query Retrieval and Web Page Categorization Using Neural Networks, in Division of Computer and Systems Sciences, Gradute School of Engineering, vol. Doctoral. Osaka: Osaka Prefecture University , pp. 94.

[2] R.A. Calvo, M. Partridge, and M. A. Jabri, 1998. A Comparative Study of Principal Component Analysis Techniques, presented at In Proc. Ninth Australian Conf. on Neural Networks, Brisbane P. Frasconi, G. Soda and A. Vullo. "Text categorization for multi page document: a hybrid naive Bayes HMM approach", In proceeding of 1st ACM/IEEE-CS joint conference on Digital libraries; ACM Press New York, NY, USA, pages 11-20. 2001

[3] A.M. Kibriya, E. Frank, B. Pfahringer and G. Holmes. "Multinomial naive bayes for Text categorization" revisited. AI 2004: Advances in Artificial Intelligence, 3339, pp. 488–499, 2004.

[4] G. D. Guo, H. Wang, D. Bell, Y. X. Bi, and K. Greer."Using kNN model for automatic text categorization". Soft Computing, 10(5), pp. 423–430, 2006.

[5] R. N. Chau, C. S. Yeh, and K. A. Smith. :"A neural network model for hierarchical multilingual text categorization". Advances in Neural Networks, LNCS, 3497, pp. 238–245, 2005.

[6] S. Gao, W. Wu, C. H. Lee, and T. S. Chua. "A maximal .gure-of-merit (MFoM)-learning approach robust classifier design for text categorization". ACM Transactions on Information Systems, 24(2), pp. 190–218, 2006.

[7] R. Schapire, Y. Singer, and A. Singhal. "Boosting and Rocchio applied to text clustering". In Proceedings of the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, pp. 215–223, 1998.

[8] Broder, M. Fontoura, E. Gabrilovich, A. Joshi,V. Josifovski, and T. Zhang. "Robust classi.cation of rare queries using web knowledge". In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, pp. 231–238, 2007.

[9] Z. Cataltepe and E. Aygun. "An improvement of centroid-based classi.cation algorithm for text classification". IEEE 23rd International Conference on Data Engineering Workshop, 1-2 pp. 952–956, 2007.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 06, June-2021

[10] D. Lewis and J. Catlett. “Heterogeneous uncertainty sampling for supervised learning”. In Proceedings of the Eleventh International Conference on Machine Learning, pp. 148–156, 1994.

[11] Dumais and H. Chen. “Hierarchical classification of Web content”. In Proceedings of the 23rd Annual International ACM SIGIR conference on Research and Development in Information Retrieval, Athens, Greece, pp. 256–263, 2000.

[12] David D.Lewis, Robert E. Schapire, James P. Callan, nad Ron Papka. “Training algorithms for linear text classifiers”. IN SIGIR’96: Proceeding of the 19th Annual International AGM SIGIR Conference on Research and Development in Information Retrieval, pp.298-306,1996.

[13] Makato Iwayama and Takenobu Tokunaga. “Cluster based text categorization:a comparison of category search strategies”. In proceedings of the 18th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR’95), pp. 273-281,1995.

[14] D.D. Lewis, “Naive (bayes) at forty: The independence assumption in information retrieval”. In 10th European Conference on Machine Learning (ECML-98), pp. 415, 1998.

[15] A.McCallum and K. Nigam. “A comparison of event models for naive bayes text classification”. In AAAI-98 Workshop on Learning for Text Categorization, 1998.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)