Assignment 1

Sebastian Bengtsson
DVAMI20h
BTH
Karlskrona, Sweden
sebe20@student.bth.se

Marius Stokkedal
DVAMI20h
BTH
Karlskrona, Sweden
matk20@student.bth.se

I. Introduction

Given a dataset with the usage of words from e-mails, preselected as spam or non-spam, the task was to pre-process the data by turning the continuous values of the dataset into discrete values that could be used with the implemented LGG algorithms, and then calculate the accuracy of the classification.

II. Data preprocessing

A. The original dataset

The dataset consisted of 58 columns, the first 48 of them being continuous values [0,100] representing the percentage of how much the specific word showed up in that e-mail. The next 6 columns were the same but for characters instead, such as "#". The next 3 columns were attributes with information of how much capital letters were used in the e-mail.
The final column was a nominal {1,0} class attribute that showed if the e-mail was considered a spam mail or not.

B. Discretization

Discretization of the values was carried out because it would provide fewer buckets for each attribute, and therefore a more general definition of an e-mail. A large part of the values were 0s, since most e-mails did not contain any instance of most of the words. The rest were still very close to 0, as it was a percentage rate.

A lot of methods were tried, for example setting the values to either 1 or 0 after comparing them to specific fixed values or the mean of the columns and rows but it gave too few values to be used with the LGG algorithms (no column contained 1 in every instance with gave an empty LGG).

At this point a large training set was used (~25%) which led to the result that it was not possible to find an attribute that was in every one of those instances post-discretization. The solution that was found was to be much more lenient when binning the words, e.g., a low threshold for binning a word as 1. This gave us results which we were able to use with the LGG.

III. Calculations

A. Size of possible instances

Possible instances are how many forms an instance in the data can take. To calculate this, the possible number of values for every attribute is multiplied together. [1, p. 106] If the data has n attributes (A) the number of possible instances (I) is calculated as

$$\prod_{i=0}^{n} A_i = I. \quad (1)$$

Since all the data was converted to binary form, the number of possible values for each attribute is 2 (0 and 1) and there were 57 different attributes. This gave the possible instances to be $I \approx 1.44 \cdot 10^{17}$ (1).

B. Size of hypothesis space

The hypothesis space (H) is a measure the number of outcomes (O) based on the possible inputs (I) [1]; and is calculated as

$$O^I = H. \quad (2)$$

This gives $H = 2^{2^{57}} = 2^{114} \approx 2.08 \cdot 10^{34}$. (2)

C. Hypothesis space of conjunctive concepts

Treating the absence of a feature as an additional value, we can calculate the hypothesis space of conjunctive concepts as

$$\prod_{i=0}^{n} A_i + 1 = C. \quad (3)$$

This gives C $\approx 1.57 \cdot 10^{27}$. (3)

D. Results

5 tests were run to get an average accuracy from the model (Table 1).

TABLE I. Accuracy

Run #	*True positive*	*False positive*	*True negative*	*False negative*	*Accuracy*
1	761	104	2684	1047	75%
2	1603	1619	1169	205	60%
3	824	123	2665	984	76%
4	1360	592	2169	448	77%
5	761	104	2684	1047	75%
				Average:	***72.6%***

An average of 72.6% shows that the model can somewhat predict if an e-mail is spam or not. The total amount of FP was 2542 vs FN total of 3731. This shows that when it went wrong the model was less likely to class a non-spam as spam than it was to class a spam as non-spam, which was the desired result.

IV. Impletmented algorithms

A. LGG-Set

The aim of this algorithm is to find the attributes which are present in all instances of the data. When implementing this in the python code a pandas data frame is sent in and then iterated through row for row. LGG-Set is used together with the function LGG-Conj and the arguments is then the current row and the conjunction row which LGG-Conj produces.

B. LGG-conj

This algorithm takes in two rows from the data frame in LGG-Set and iterates through every column and marks every instance of both rows having TRUE, in this case represented by 1, in the same attribute. The function then returns the row with the conjunction between the two rows.

V. References

[1] P. Flach, Machine Learning - The Art and Science of Algorithms that Make Sense of Data, Cambridge University, 2012.