Assignment 1

1st Abdulkarim Dawalibi
2nd Adnan Altukleh

I. Introduction

The number of spam emails received in our inboxes has slightly increased in the last decade [1]. For this reason, we are going to train a model to predict if an email is a spam or not.

We are going to use the least general generalization algorithm to train our model. The dataset will be used to solve the problem called 'Spambase Data Set' which consists of 57 attributes and 4601 instances [2].

II. Method

A. Data cleaning

Started by checking for missing values and detecting outliers. Also dropped all existing duplicated instances.

B. Preprocessing

In this step, we class discretized the data using binning with help of the Sklearn KBinsDiscretizer library. We used the kmeans strategy, decided to make 8 bins and encode the data ordinal. We fitted the data and then transformed the data [4].

C. Computing the instances and hypotheses space and conjunctive concept

To compute the instance space, we count how many unique values each attribute has, and then multiply these values to get the instance space [3].

Hypothesis space is the number of possible extensions which means the number of output classes power of the instance space [3].

The number of conjunctive concepts is adding the absence of a feature value for each attribute [3].

D. Implementing the algorithm

We used pseudo code 4.1 and 4.2 from the literature book [3] to implement our code. Instead of dropping the non-common feature when algorithm 4.2 was implemented, we decided to give the non-common features value -1.

E. Train the model and evaluation

The data were divided into two subsets train data and test data. The train data contained 80% of the records where they were classified as spam. While the test data contained 20% of the spam records and contained the same number of none spam records of the whole data set excluding the trained records.

After training the model, we get an instance that contains the common attributes with their values. The non-common attributes have a value of -1. Then we compared the instances with the test data to get the indices of the identical instances of "matched rows". Afterwards, we created a new column called "predicted". The instances that were not classified as spam got a value of 0 in the predicted column excluding the matched instances which got a value of 1.

To evaluate the results, we used a confusion matrix, accuracy score and precision score implemented with the help of Sklearn.metrics.

III. Results

As a result of data cleaning, we got 391 duplicated instances otherwise, no outliers nor missing values were detected.

Instance space, hypotheses space and conjunction concept results can be found in the submitted Jupyter notebook.

The conjunctive rule of our trained model consists of two features, word_freq_cs and capital_run_length_total.

The results of the trained model can also be found in the Jupyter notebook.

The accuracy score of the model varies between 49-62% depending on the training data. While the precision score of the model also varies between 56-64%.

IV. Discussion

We dropped the duplicate instances from the data set to avoid having the same instance in the train data, in this way we guarantee that our model train on the same instance more than once in this way we avoid overfitting.

In the preprocessing step we decided to use a kmeans strategy and 8 bins because the values in each bin have the same nearest center of a 1D k-means cluster. If we tune the parameters such as bin-size and the strategy, we get lower accuracy. Because of the higher number of common features.

That suits our model and our dataset because the interval between values does not have a big gap but in the last three columns, the gap is bigger. We checked the median, mean, max and min values for each column.

For most of the columns, the median is 0. Because if we have many bins this affects our data set if we add many bins, we then get many large intervals in each bin which negatively affects our results and does not gain anything from the Discretization.

In the 4.2 algorithm instead of dropping the non-common features we choose to give them a value of -1 to make it easier for us to compare between all the instances in the test data.

We decided to train the model on spam emails because it gave us better accuracy results than training the model on none spam emails.

When we reduce the size of the bin, we get more common features, after training the model this cause that the model performs worse in the testing process.

The test data contained equal number of instances of both spam and none spam, because we have just 20% left of the spam data and if we use more non-spam than the spam it will

affect the accuracy better, but the model precision will decrease.

V. Conclusion

The kmeans strategy and bin size of eight, suit our algorithms best, to give the best accuracy score and precision score. As we tested, we realized that the number of bin size affects the conjunctive rule. The lower bin sizes are the higher the conjunctive rule gets. There are more none spam instances than spam instances in the data set, that means the ratio of spam and none-spam is not equal, which might have affected our model results.

References

[1] Moorthy, J. (2022). "23 Email Spam Statistics to Know in 2022". mailmodo. https://www.mailmodo.com/guides/email-spam-statistics/

[2] Hopkins, M., Reeber, E., Forman, G., & Suermondt, J. (1999). SpambaseDataSet.UCI. https://archive.ics.uci.edu/ml/datasets/Spambase

[3] FLACH, P. (2012). MACHINE LEARNING The Art and Science of Algorithms that Make Sense of Data. CAMBRIDGE UNIVERSITY PRESS. Pp. 136-140.

[4] Brownlee , J. (2020). How to Use Discretization Transforms for Machine Learning. Machine Learning Mastery. https://machinelearningmastery.com/discretization-transforms-for-machine-learning/