**CSC 430 Project**

# **Email filter for Uninteresting university e-mails using K-NN and Naïve Base**

*Ghada Raee 19195*

*Noor Aljabban 18630*

Content

* Abstract
* Problem Description
* K-NN Description
* Naïve bayes Description
* Data
* Conclusion
* References

Abstract

This project regards a specific problem observed in GUST university by the students. Students receive many emails from many departments, but recently the number of uninteresting emails from clubs and certain departments has been increasing, overshadowing the important emails. Thus, we created a filter for unimportant emails. We went about two methods of implementing this filter, using K-NN algorithm and naïve bayes algorithm. naïve bayes is more commonly used as an email filter, but after implementing both algorithms, we have observed that the results were only slightly different in accuracy, though naïve bayes had better time complexity and was the most accurate.

Problem Description

In Gulf University for Technology and Science, students receive many emails from various university departments, some of which are important or interesting enough for the student, like emails from their professors, and emails regarding registration or prizes or such, however, there has been a flood of emails by clubs and other departments regarding less important events. Since these emails are increasing in frequency, they have gotten in the way of students to view the important emails. Therefore, to make sure that this is a relevant problem for GUST students, we have conducted a survey that asks the students the types of emails they find uninteresting from various departments with the ability to choose multiple answers, secondly asking them if they find club emails annoying, and lastly asking them if they find club emails to be too many at a time. The results of the survey did confirm what we have predicted about the students’ opinions regarding some emails being unimportant or abundant.

The survey questions were as follows:

Chart, bar chart

Description automatically generated

This shows that students find club emails and emails from the athletic department to be the least important, even with such a small sample that fact was clear.

Chart, bar chart

Description automatically generated

This shows that there is a demographic that is not necessarily interested in emails from clubs specifically, which we took into consideration when we put the training data.

Chart

Description automatically generated

Lastly, This illustrates that more than ¾ of students from our random sample think that clubs spam students with emails.

K-NN Description

This is a method that approaches categorization of new data using similarity and distance instead of a generalized rule that categorizes new data based on a rule derived from training data. Depending on similarity means that the aim is to take the learning data and represent it in multidimensional feature space, then the rule of categorization becomes, the smaller the distance between an entered data and the training data in the feature space, the more two examples are similar, thus they belong in the same category. In the K-nearest neighbour method, K points are taken instead of just one, so that the class can be determined based on the most members similar among the K-nearest neighbours.

Naïve Bayes Description

Naïve bayes is a type of simple Bayesian network. The concept is that there is a centre node that has dependencies with all the other features (nodes) in a system. The simplification comes from not having any dependencies between the features themselves. Naïve bayes is based on a probabilistic model that predicts the category of an item based on conditional probability that calculates the probability that a new item that gets added belongs to a certain class from the main node (classification node), given the other features (nodes) that are connected to that main node.

An example of how naïve bayes will be modelled for each email is:

*The email text:* “Dears, Assignment 2 is posted on the portal. It is due on Thu 24th of Feb 2022. Regards. See this post in context”

*the model: Diagram

Description automatically generated*

Data

We have collected a sample from the emails sent on the gust email to students and put them in an excel file Emails2.xlsx that contains 200 emails sent to GUST students with the classification of which emails are interesting and which are not. The data was categorized based on the results of the survey. Each record contains the following categories:

* Email text content
* Classification (true for interesting, false for uninteresting)

Methodology

The data used in this project was 200 gust emails labelled with interesting (True) and not interesting (False) based on the survey that we conducted to know which emails are interesting to gust students and which are not. Before applying the algorithms on each email, the emails were turned into lowercase letters and were cleaned by removing punctuation, emojis and non-keywords like articles, connectors …etc. Then, that data was divided into two parts in which 80% of the data were used for training and 20% were used for testing.

The first algorithm that was applied on the data was Naïve Bayes. In Naïve Bayes the training data were split according to their labels in 2 separate lists (interesting emails list and non-interesting email list). From each list, a dictionary of vocabulary was created, for example, if the email was about a seminar and it included the following words: ‘seminar’, ‘location’, ‘time’…etc, these words were added to the non-interesting vocabulary dictionary and the same goes for interesting emails, a dictionary of interesting emails vocabulary was created. Then, each word in each dictionary was counted in its related emails, in other words, the word ‘seminar’ was in the non-interesting vocabulary dictionary, ‘seminar’ was counted in all non-interesting emails and added to another dictionary with its frequency in non-interesting emails and the same process was applied to every other word in non-interesting vocabulary. The same code was applied to interesting vocabulary as well.

Regarding the testing, after reading the test data and cleaning it as mentioned above, each email was split into list of words and any word that was not seen in the training vocabulary (both interesting and non-interesting vocabulary) was removed. Then, the following formulas were applied on each email:

I: interesting, w: [interesting/ not interesting] word, w1…wn : [interesting/ not interesting]words in single email [interesting email or not interesting email]

and were calculated from the training data by counting interesting emails and dividing the number on the total number of training emails, while the was counted by (1- ). was got as mentioned above by counting the word in all interesting emails (Same for ). . To avoid getting a zero in case the word did not exists in one of the 2 vocabulary dictionaries, 1 was added to both the numerator and denominator. After all the calculations were done, if turned out to be greater than then the email would be classified as interesting, otherwise it will be classified as not interesting.

The second algorithm that was applied is K-NN. To pave the way for the algorithm, each word in each email was counted (counted the frequency of word in the email itself not in all emails) and the word with its frequency was added to a dictionary. The counting was done on each email and at the end all dictionaries were gathered together in a list. The same process was done on the test data. After that, the Euclidean distance was calculated for each test with each training email. (so the Euclidean distance of TestEmail1 with TrainingEmail1, TrainingEmail2… TrainingEmailN and then TestEmail2 with TrainingEmail1, TrainingEmail2… TrainingEmailN and so on). The similarity was measured by the frequency of each word of the testing email compared to the frequency of the same word in the training emails one-by-one to find the most similar emails to the test email, and that was the base for the calculation of Euclidean distance. Each distance for test email was added to list and that list was sorted, then the first K similarities were taken (K will be discussed in the following paragraph). Finally, if the test email was similar to more interesting emails than non-interesting then it will be classified as interesting email, otherwise it will be classified as not interesting.

K was chosen based on the accuracy of the results. The algorithm was applied on different odd Ks and the one that had the most results that were the most accurate were chosen. When K=35, the results were most accurate so, the chosen K was 35.

The accuracy for both algorithms was calculated by dividing the number of correct classifications over the total number of test data. The calculations showed that Naïve Bayes approach is slightly more accurate than K-NN. Naïve Bayes had accuracy of 85 % and got 34 emails classified correctly out of 40 while KNN was accurate by 82.5% and got 33 emails classified correctly out of 40. In other words, there no big difference in the results in both algorithms. However, the time needed to for execution was calculated for and Naïve Bayes approach was always faster than K-NN approach which makes it much better than K-NN.

In naïve bayes all the operations cost O(E\*EW) where E stands for number of emails (testing emails or training emails) and EW stands for the amount of words each email, except for multiplying the probabilities which costs O(P) where P stands for the number of probabilities, so the time complexity is O(E\*EW). While in K-NN, if the K was entered manually the time complexity will be O(ETe\*ETr\*TeW) where ETe stands for the size of testing emails, ETr stands for training emails and TeW stands for unique words in a testing email. This time complexity is due to the need to find the most similar email in training data for a testing email using Euclidean distance. However, if K was set automatically by the program, the same algorithm will be applied n/2 times (which will be 25 in our example) to find the best K among 25 odd number, so, it will be much slower.

Outputs:

Naïve Bayes:

A picture containing text

Description automatically generated

KNN with getting K automatically:

Text

Description automatically generated with low confidence

KNN with entering K manually:

A screenshot of a computer

Description automatically generated with medium confidence

Conclusion

In conclusion, it has been observed that naïve bayes is more accurate in the classification of uninteresting emails, with its level of accuracy being 85%, which is a relatively high accuracy rate, and that is most likely because naïve bayes analyses the contents of the emails and then categorizes it based on the frequency of when this content was interesting, thus, it achieved slightly higher levels of accuracy in comparison with K-nearest neighbour that categorized based on similarity of the overall content among testing data and training data. K-nearest neighbour’s level of accuracy was the same both when K was manually entered after being calculated and when it was automatically set, however, what was a big difference is the time of execution that was significantly higher when K was set automatically. Regardless, despite how little K-NN’s time of execution was when K was set manually, naïve bayes was much faster in comparison, which was very noticeable. Therefore, overall, even if K-NN preforms good in filtering emails, naïve bayes remains remarkably better, faster, and slightly more accurate.

References

W. Ertel. *Introduction to Artificial Intelligence*, second edition, Springer, 2017. ISBN: 9783319584874.