Natural Language Processing

Mandatory Problem
Coursework Problem 7.3

Members	**Matriculation Number**
Daria Likhacheva	00754308
Muhammad Daniel Bin Mohd Khir	00818571
Olha Solodovnyk	00818239
Sofia Gutoranska	00819789

Examiner
Prof. Dr. Patrick Glauner

Introduction

More than 85 billion spam emails are sent out every single day. [1] The amount of manual effort that would be required to filter through these emails is unimaginable. Thankfully, automatic spam filters in our email clients do the heavy lifting for us.

But what is considered spam? We define spam as unsolicited, unwanted emails that may contain potentially harmful programs. A spam filter is a program that is able to examine each email and determine if the email message is spam.

In this report, we outline a basic implementation of a spam filter in Python with the naive Bayes classifier. Our goal is to create a spam filter which has an accuracy of at least 80% when classifying messages.

Methodology

In this section, we explain the mathematical and conceptual foundations of the spam filter.

Bayes Rule

Bayes' rule describes the probability of an event based on known prior events related to it.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$Posterior = \frac{Likelihood\ x\ Prior}{Normalizer}$$

Bag of Words

A bag of words is a table of all the words occurring in a corpus along with their frequencies.

Given the corpus of the following phrases:

1. CLICK SECRET LINK
2. SECRET SPORTS LINK
3. PLAY SPORTS TODAY
4. SECRET SPORTS EVENT

The bag of words is as follows:

Word	Frequency
CLICK	1
SECRET	3
LINK	2
SPORTS	3
PLAY	1
TODAY	1
EVENT	1

These two concepts (Bayes' rule, bag of words) are the foundations of our naive Bayes classifier.

Naive Bayes Classifier

Given a message m, classify m as either $SPAM$ or HAM (not spam).

The most likely class of a message, $C \in (SPAM, HAM)$ is chosen with:

$$C = \underset{C}{argmax}\, P(C) \prod_{i} P(x_i|C)$$
$$where\ x\ is\ a\ word$$

Remarks:

1. Assumption: Position of words in the text does not matter.
2. Assumption: The probabilities $P(x|C)$ are conditionally independent.
3. The normaliser is dropped as it is independent of C.

We feed the bag of words into a naive Bayes classifier. The probability of each word occurring in spam or non-spam messages is calculated as follows:

$$P(word|C) = \frac{count(word\ in\ C)}{sum(all\ words\ in\ C)}$$

To avoid overfitting, we use Laplacian smoothing:

$$P(word|C) = \frac{count(word\ in\ C) + 1}{sum(all\ words\ in\ C) + 1 \times (sum(all\ words\ in\ C))}$$

These probabilities are then collected and fed into the classifier for a new message.

Results and Discussion

In this section, we discuss the usage and performance of the spam filter.

Dataset

To test the spam filter, we used the Youtube Spam Collection Dataset from the UCI Machine Learning Repository. [2] The dataset contains 1956 messages extracted from 5 YouTube videos.

The dataset contains the following columns:

Column	Description
COMMENT_ID	Identifier for a comment.
AUTHOR	Username of the comment author.
DATE	Timestamp when comment was posted.
CONTENT	Actual message of the comment.
CLASS	Type of comment. (SPAM = 1, HAM = 0)

The dataset was cleaned and split into randomised train/test sets with a 70:30 ratio.

Results and Metrics

The results of classification on the test set was as follows:

	Actual Spam (1)	Actual Ham (0)	Sum
Predicted Spam (1)	281	29	310
Predicted Ham (0)	15	261	276
Sum	296	290	586

The metrics were as follows:

$$Accuracy = 0.9249$$
$$Precision = 0.9064$$
$$Recall = 0.9493$$
$$F_1\ Score = 0.9273$$

Strengths and Limitations

As evidenced by the metrics above, our model can reliably exceed our goal of prediction accuracy of 80%. It also exhibits a high precision and recall value, which means the model is able to reliably return many correct predictions.

However, the predictions are only limited to a particular type of spam: YouTube comments. We propose that the model would not perform as well when applied to other types of spam, such as SMS or email spam.

Furthermore, unknown words are not handled properly by the current model. These are words that do not occur in the bag of words used during initial training. If the model is asked to classify a message with many unknown words, it will not deliver expected results.

Lastly, the model only functions with spam comments in English. A different dataset would be required if the model is to be used to classify comments in other languages.

Conclusion

In this report, we demonstrated the need, usage, and performance of a spam filter implemented in Python. We also briefly described the mathematical and conceptual foundations of the spam filter.

Possible improvements to the spam filter would be to train it on a larger dataset. Alternatively, a different classifier could be used that provides a greater accuracy, such as tf-idf.

References

[1] "Global average daily spam volume 2021," *Statista*. https://www.statista.com/statistics/1270424/daily-spam-volume-global/ (accessed Jun. 08, 2022).

[2] "UCI Machine Learning Repository: YouTube Spam Collection Data Set." https://archive.ics.uci.edu/ml/datasets/YouTube+Spam+Collection (accessed Jun. 08, 2022).