Spam Detection
Naïve Bayes and Logistic Regression

M. Hooghiemstra
i6119992

F. Nouwens
i6166905

1 Introduction

Nowadays, people get a lot of emails during the day and also have to read through them to extract the provided information. Fortunately, the inboxes of people rarely contain spam emails anymore due to the existence of spam detection that puts the spam emails directly into the 'trash'. This is the effect of a very good spam classifier that detects spam before it can reach your eyes. The importance of spam detection and filtering makes it a well-researched area and is applied to a lot of email soft-wares (Provost, 1999). There are many detection approaches to classify an email as spam or, on the contrary, as ham. Gupta et al. (2019) mentions that most of the detection is based on classifiers that can label emails. The features that classify spam and ham are important (e.g. number of words, spelling mistakes, punctuation et cetera) within this framework.

The research question that this report answers is: *Which classifier, among the Naïve Bayes classifier (NB) and the Logistic Regression (LR) classifier, performs best on our email dataset?* To answer this research question, common evaluation metrics of the classifiers and experiments will be conducted to make the comparisons.

The reason we want to answer this question is that we have always been fascinated by the idea of spam filtering and how it actually works (and why it is so good nowadays). Since we only had one course, Natural Language Processing (NLP), that uses Python, we wanted to know how well our own classifier would do on an unseen test set with the knowledge we gained during the NLP course. This report first discusses a literature review about existing spam detection classifiers and the data used to train and test the classifiers. Then the models of the classifiers are presented. Furthermore, the report provides the experiments and the results of the classifiers. These results are discussed in the discussion section and the conclusions of the results are provided in the conclusions section.

2 Literature Review

The classification of emails is an active research area. In this section the literature on datasets and classifiers is discussed.
Bayesian classification is the most common technique used for classifying spam from ham emails. (Androutsopoulos et al., 2000) "Bayesian spam filtering learns from spam and from good mail, resulting in a very robust, adapting and efficient anti-spam approach that, best of all, returns hardly any false positives." (Tschabitscher, 2019). By finding the dataset for the spam filtering, the same website gave us an interesting and relevant paper on the Naïve Bayes classifier that compares different versions of NB. Metsis and et al. (2006) It discusses, among other things, a Flexible Bayes or Multinomial Bayes with boolean attributes and how they compare against each other. Eventually, the decision was made to not use this 'complicated' NB-classifiers in our own project, and instead focus on the simple NB, since we did not want to be too ambitious and instead wanted to learn the basic principles first (and well). Logistic Regression is also a widely-used classifier for spam detection and filtering (Englesson, 2016). Logistic regression is a baseline super-

vised machine learning tool for classification, as well as the foundation of a neural network. Lynam et al. (2006) use Logistic Regression for computing the weights for each word and then use the weighted average to predict the classification. Hereby, they minimize the cross entropy loss and this should be a good classification approach. However, Crawford et al. (2015) states that a Naïve Bayes classifier outperforms a Logistic Regression classifier on a 10-fold cross validation in classifying text. Exploring this relation between a Naïve Bayes and Logistic Regression classifier is what this project is about.

3 Data

The dataset defines spam and ham emails already. It is divided into a training set and a test set by taking two-thousand ham and spam mails as the training set, and seven-hundred-fifty ham and spam mails as the test set. This was done without any sklearn library, as the datasets are already partly pre-processed. Moreover, the dividing of the two different sets are done before the classifiers are tuned, this is done to avoid overfitting. As the number of spam and ham e-mails was kept equal, the dataset is not weighted. The encoding used for the data is 'Latin-1', which has as disadvantage that it might corrupt some data, i.e. miss some symbols (Van Rossum and Drake, 2000). However, since it is spam, this will not cause an issue as spam is already bad data.

The emails conducted from the datasets are all in separate '.txt' files, where each file has a first line detailing the subject of the message. On the next lines, the actual content of the email is shown. The message can be long, short, a forwarded email, a reply to a previous email et cetera, as long as it has a subject line.

4 Model

The models used for the classification of spam emails are Naïve Bayes and Logistic Regression. After obtaining the data, the dataset is again pre-processed, but then on the text of each email. The pre-processing steps used include tokenization, removing the stop words and getting rid of certain punctuation, but not all. When deleting errors or punctuation, it is important to keep in mind that this can change the sense of a message. Therefore, only the apostrophes, commas, periods that are not the end of a sentence, and some symbols (i.e. commas, brackets, colons et cetera) that do not express a sense of ham or spam are deleted. The 'common' spelling mistakes were kept in the emails, since they can be crucial in distinguishing spam from ham. The stop words were removed since they had no importance in classifying spam, as both spam and ham emails used them. In this way, every email obtains the same format and the same structure of sentences and words.

To model the data, for the Naïve Bayes classifier, a simple dictionary that held all the tokenized words, was created for each message. The classifier ran on this labeled dictionary. The Naïve Bayes classifier itself was imported using the sklearn package and trained on the (pre-processed) data.

The Logistic Regression classifier was also imported from the linear models of the sklearn package. The trained features and trained labels are fitted to the Logistic Regression model. Then the predictions, that take the loss function into account, are made.

5 Results

This section describes some experiments on the Naïve Bayes classifier and the Logistic Regression classifier that we conducted. The properties of the experiments are described and the results on these experiments are given in separate section of the different classifiers.

5.1 Experiments

To answer our research question, the following questions are tried to be answered by experiments:

1. Would keeping in the stop-words affect the performance of the NB classifier in regards to LR classifier?

2. Would changing the ratio of spam:ham change the performance of the NB classifier and/or LR classifier?

3. Would the rates of the evaluation metrics differ by cross-validating over the whole dataset?

The experiments were conducted in a controlled environment, where the external variables were kept the same. No external factors influenced the experimental results, and if so, they acted on all experiments, so their influence did not pose a threat to the results.
To validate the experiments more, a ROC-curve was meant to be plotted. However, this turned out to be way more difficult to implement, so in the end the decision was made to leave it out.
The third experiment that we wanted to conduct took too much time to finish in time. We left this in the list of experiments, because we think that this is an important part to experiment on.

5.2 Naïve Bayes Classifier

5.2.1 With pre-processing

This section regards the first experiment, see Experiment section, by evaluating the NB classifier with pre-processing.
Accuracy = 97.46666666666666% ≈ 97.47%
Below are the values obtained for the precision, recall and F-score (all approximated). The discussion of the results is in section 6.

	Precision	Recall	F-score
Macro[1]	97.56%	97.47%	97.48%
Micro[2]	97.47%	97.47%	97.47%

Table 1: Results Naïve Bayes classifier with pre-processing

Below the confusion matrix of the NB classifier is depicted. These values are all out of the seven-hundred-and-fifty test emails.

	Actual Results	
Classifier	Ham	Spam
Ham	357	18
Spam	1	374

Table 2: Confusion matrix of Naïve Bayes classifier with pre-processing

5.2.2 Without pre-processing

This section regards the first experiment, see Experiment section, by evaluating the NB classifier without pre-processing.
Accuracy = 98.0%
Below are the values obtained for the precision, recall and F-score (all approximated).

	Precision	Recall	F-score
Macro[3]	98.06%	98.00%	98.00%
Micro[4]	98.00%	98.00%	98.00%

Table 3: Results Naïve Bayes classifier without pre-processing

The discussion of the results is in section 6.
Below the confusion matrix of the NB classifier is depicted. These values are all out of the seven-hundred-and-fifty test emails.

	Actual Results	
Classifier	Ham	Spam
Ham	361	14
Spam	1	374

Table 4: Confusion matrix of Naïve Bayes classifier without pre-processing

5.3 Logistic Regression

5.3.1 With pre-processing

This section regards the first experiment, see Experiment section, by evaluating the NB classifier with pre-processing.
Accuracy = 94.85396383866481% ≈ 94.85%
Below are the values obtained for the precision, recall and F-score (all approximated).

	Precision	Recall	F-score
Macro	95.07%	94.95%	94.97%
Micro	94.85%	94.85%	94.85%

Table 5: Results Logistic Regression classifier with pre-processing

From the table, it can be seen that macro has a higher score for precision, recall and F-score than for micro.
Below the confusion matrix of the LR classifier is depicted. These values are all out of the seven-hundred-and-fifty test emails.

[1] Macro means that it calculated metrics for each label, and found their unweighted mean.

[2] Micro means that it calculated metrics globally by counting the total true positives, false negatives and false positives.

	Classifier	
actual results	Correct	Not correct
Selected	12	61
Not selected	7	53

Table 6: Confusion matrix of Logistic Regression classifier with pre-processing

5.3.2 Without pre-processing

This section regards the first experiment, see Experiment section, by evaluating the NB classifier without pre-processing.
Unfortunately, this experiment could not be done by the way the code is structured.

6 Discussion

The results for the Naïve Bayes classifier were higher than expected, since nowadays there are many ways for spammers to reduce the effectiveness of NB and LR. For the Naïve Bayes classifier some examples are Bayesian Poisoning or using pictures that replace text (Sprengers, 2009). Bayesian Poisoning is when an email contains a lot of valid text that could very well be in a ham-email, but still is spam.

The difference in performance NB with pre-processing and without pre-processing (both still use tokenization) is highly correlated with the fact that ham-emails are often way more personal and thus use more personal pronouns (i.e. you, me, we, us et cetera). Removing these kind of words will make the email more generic and thus more likely to look like spam. This supports the higher accuracy and scores for NB without pre-processing.

For the Logistic Regression classifier, it performs worst when there are more values in the diagonal left-bottom to top-right (Shetty, 2018). LR is also a bit influenced by Bayesian poisoning. However, since the data samples are from 2005 (Metsis and et al., 2006), these newer techniques were not yet implemented in the used dataset so the Naïve Bayes classifier was still very good. This can be concluded on the fact that the accuracy is similar to the precision, recall and F-score.

Earlier literature on comparing the Naïve Bayes classifier with the Logistic Regression classifier already suggested that the former would give the best results. In section 5 Results it can be seen that for both macro and micro, the precision, recall and F-score are better for the Naïve Bayes classifier for experimenting on datasets with pre-processing than for the Logistic Regression classifier. However, according to Table 6 from the previous section, the results on the Logistic Regression are not accurate since the sumed rates for the confusion matrix do not sum up to the size of the test set (750). Multiple attempts were made to adjust the LR-classifier so that it worked on the train and test data, however, this turned out to be infeasible within the given time-frame. In the end, we left the 'digits.data' in, to verify if logistic regression actually worked.

7 Conclusions

We discussed and evaluated the implemented classifiers in 'detecting' spam emails. The desired outcomes were not reached due to the fact that the Logistic Regression classifier was not implemented in a correct way. This lead to unfinished experiments on the LR classifier, as well as no comparison-experiments between the two classifiers.
Nonetheless, the Naïve Bayes classifier itself gave some very good results, that could be explained due to the simplicity of the dataset. Hence, we can only give a conclusion on the Naïve Bayes classifier, namely that pre-processing on stopwords does not improve performance and NB is a very good baseline for 'simple' spam emails.

From this project, not many new things can be learned in regards to the well-researched topic of Naïve Bayes in spam detection classifiers. Nonetheless, we are still pleased with how our own NB-classifier turned out.

References

Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, and Constantine D. Spyropoulos. 2000. An experimental comparison of naive bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '00, pages 160–167, New York, NY, USA. ACM.

Nikhila Arkalgud. 2008. Logistic regression for spam filtering.

Michael Crawford, Taghi M Khoshgoftaar, Joseph D Prusa, Aaron N Richter, and Hamzah Al Najada. 2015. Survey of review spam detection using machine learning techniques. *Journal of Big Data*, 2(1):23.

Niclas Englesson. 2016. Logistic regression for spam filtering. Bachelor's thesis, Matematiska institutionen, Stockholm University, June.

Yoav Goldberg and Graeme Hirst. 2017. *Neural Network Methods in Natural Language Processing*. Morgan & Claypool Publishers.

Vashu Gupta, Aman Mehta, Akshay Goel, Utkarsh Dixit, and Avinash Chandra Pandey. 2019. Spam detection using ensemble learning. In *Harmony Search and Nature Inspired Optimization Algorithms*, pages 661–668. Springer.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing (2Nd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Thomas R Lynam, Gordon V Cormack, and David R Cheriton. 2006. On-line spam filter fusion. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 123–130. ACM.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA.

Vangelis Metsis and et al. 2006. Spam filtering with naive bayes – which Naive Bayes? In *Third Conferences On Email And Anti-Spam (CEAS)*.

Thomas M. Mitchell. 1997. *Machine Learning*, 1 edition. McGraw-Hill, Inc., New York, NY, USA.

Jefferson Provost. 1999. Naıve-Bayes vs. rule-learning in classification of email. *University of Texas at Austin*.

Badreesh Shetty. 2018. Supervised machine learning: Classification. https://towardsdatascience.com/supervised-machine-learning-classification-5e685fe18a [Online; accessed 26-May-2019].

Martijn Sprengers. 2009. The effects of different bayesian poison methods on the quality of the bayesian spam filter spambayes.

Heinz Tschabitscher. 2019. What you need to know about bayesian spam filtering. https://www.lifewire.com/bayesian-spam-filtering-1164096. [Online; accessed 26-May-2019].

Guido Van Rossum and Fred L Drake. 2000. *Python reference manual*. iUniverse.

Appendices

A Personal Thoughts & Future Ideas

The project was a lot harder than we expected. There were a lot of issues with the coding in Python, including but not limited to loading the data, obtaining the right formats, training the classifiers et cetera. Both group members spent a lot of time on the project, but in the end we did not get the desired outcome. Things we could not do but wanted to:

- **Custom Word Filtering approach**: the idea was to implement a classifier that would classify a message based on certain words, punctuation or phrases that occurred in the messages.
- **Experimenting on the second and third experiment in section Experiments**: Having the LR-classifier work well, so that the results of the experiment would be valid.
- More figures and graphs that support the results on experiments

B Project Proposal

B.1 Plan

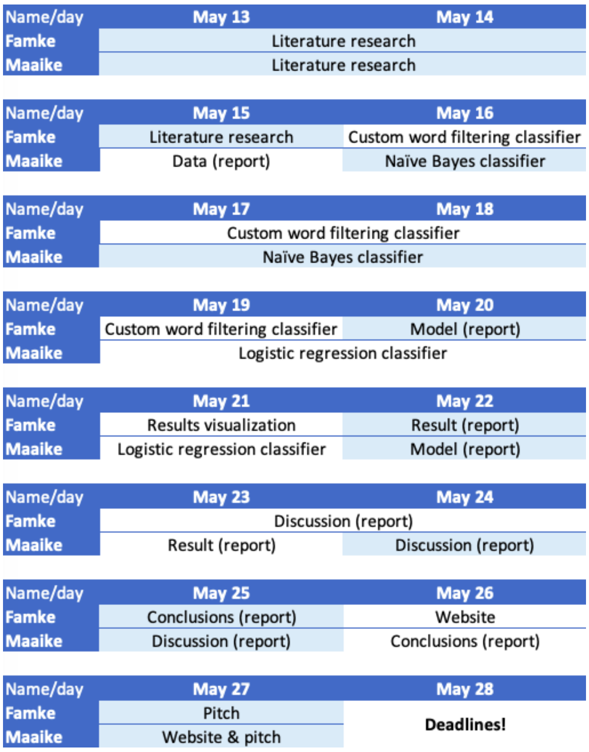
The concrete plan of action is depicted in Appendix A in Figure 1.

B.2 Expected Outcome

The outcomes will all depend on the classifiers. The Naïve Bayes classifier is a good baseline for text classification. We think that these classifiers will be relatively good classifiers due to the research that we did on them. This is seen in the Literature review section.

C Project Planning

See Figure 1.



Name/day	May 13	May 14
Famke	Literature research	
Maaike	Literature research	

Name/day	May 15	May 16
Famke	Literature research	Custom word filtering classifier
Maaike	Data (report)	Naïve Bayes classifier

Name/day	May 17	May 18
Famke	Custom word filtering classifier	
Maaike	Naïve Bayes classifier	

Name/day	May 19	May 20
Famke	Custom word filtering classifier	Model (report)
Maaike	Logistic regression classifier	

Name/day	May 21	May 22
Famke	Results visualization	Result (report)
Maaike	Logistic regression classifier	Model (report)

Name/day	May 23	May 24
Famke	Discussion (report)	
Maaike	Result (report)	Discussion (report)

Name/day	May 25	May 26
Famke	Conclusions (report)	Website
Maaike	Discussion (report)	Conclusions (report)

Name/day	May 27	May 28
Famke	Pitch	**Deadlines!**
Maaike	Website & pitch	

Figure 1: The plan for the project per day per person.