DAT405/DIT406 Introduction to Data Science and AI
Assignment 4: Spam classification using Naïve Bayes

Anh Thu DOAN
Exchange student from CY Tech - France to GU CSE Department
12 hours spent

Khushi Chitra Uday
Exchange student from CY Tech - France to GU CSE Department
10 hours spent

Group 26

April 27, 2022

1 Introduction

Spam (or Spam Email) is one of the enormous problems in today's technology world. It causes many inconveniences for users. This is the way a lot of companies are abusing to communicate with customers. In this assignment, we will use the data from SpamAssassin public mail corpus, a selection of mail messages, suitable for use in testing spam filtering systems. The main purpose of this project is to implement a Naïve Bayes classifier in Python that will classify emails into spam and non-spam ("ham") classes. Naïve Bayes Classifier is one of the simplest and most efficient classification algorithms that helps to build fast predictive speed machine learning models.It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.

2 Materials

We downloaded 3 different files which are:

- easy-ham: non-spam messages typically quite easy to differentiate from spam messages.
- hard-ham: non-spam messages more difficult to differentiate.
- spam: spam messages.

form https://spamassassin.apache.org/old/publiccorpus/.

3 Results

3.1 Preprocessing

We used sklearn.datasets.load_files to load files with categories as subfolder names in pairs such as easy-ham & spam, hard-ham & spam. The data now stored in Bunch, which include data, became a list of string, and the target was ndarray with the target labels (spam is stored as 1 in target, easy-ham and hard-ham are assigned for 0).

In the data we used, there are 2551 easy-ham emails and 501 spam emails in the easy-ham spam data set. So that there are more than 80% of the easy-spam data set is non-spam messages. On the contrary, there are only 250 hard-ham emails were used in the hard-spam data set even we used the same number of spam emails. Therefore, there are only 33% of the data set are non-spam messages.

We split the spam and easy-ham, spam and hard-ham in a training set and test set so we would not train and test on the same data.
Sklearn.model_selection.train_test_split was used to split our data into training

set(70%) and test set (30%) (we set test_size = 0.3). For all cases below we take into account the precision and recall for only the positive class, i.e spam.

3.2 Functions to run Naïve Bayes Classifier

Functions with the following name and arguments:
def run_naiveBayes(ham_train, ham_test, spam_train, spam_test):

In this task, we created two functions to run two different types of Naïve Bayes Classifier in SKlearn, which are Multinomial Naïve Bayes (MultinomialNB()) and Bernoulli Naïve Bayes (BernoulliNB()). In the first step in the function, we used CountVectorizer in sklearn.feature_extraction.text to learn the vocabulary dictionary and return document-term matrix for the ham_train (**fit.transform(ham_train)**) and only transform documents to document-term matrix for ham_test (**transform(ham_test)**).

As we mentioned above, we used two different types of Naïve Bayes Classifier in our two functions. The Multinomial Naive Bayes models the number of count of the feature occurs (in this case it is the word), while the Bernoulli models base on the presence/absence of the feature (word). For the case of Bernoulli Naive Bayes we used the parameter binarize to make the features binary. The default value for binarize is 0, so that any count over 0 will be mapped to 1/True.

3.3 Run the functions

3.3.1 Spam versus easy-ham

3.3.1.1 Multinomial Naive Bayes Classifier

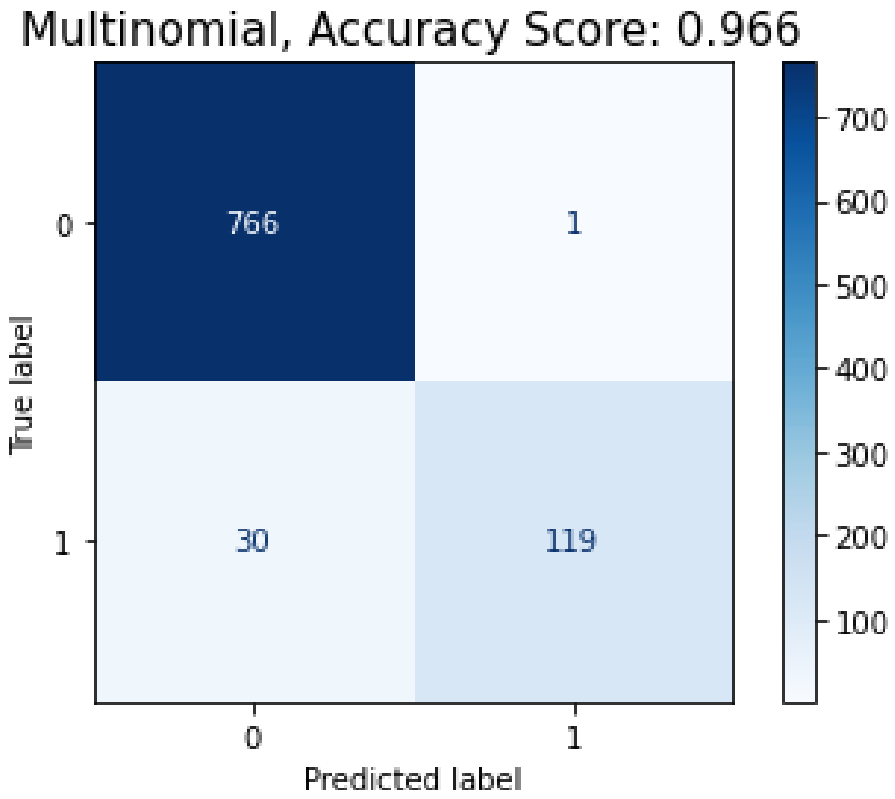

Figure 1: Spam versus easy-ham on Multinomial Naive Bayes

- Multinomial, Precision score: 0.99167
- Multinomial, Recall score: 0.79866

Here even though the model has a high accuracy and precision score it has a low recall which means it misclassifies spam emails as easy ham(false negative = 30). The high precision score means that we have a low misclassification of easy-ham emails which for us only happens once. The multinomial classifier as a high accuracy score of 0.966 when classifying easy-ham and spam which can easily be misleading when the classification problem is given skewed data. In these cases we have to use other matrix such as recall and precision.

3.3.1.2 Bernoulli Naive Bayes Classifier

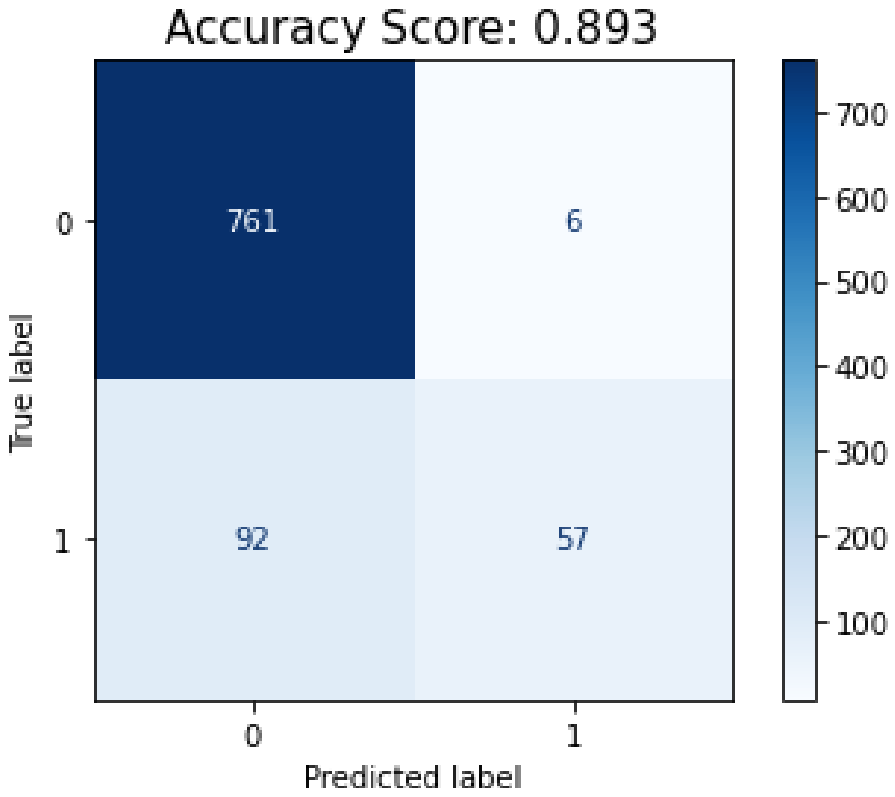

Figure 2: Spam versus easy-ham on Bernoulli Naive Bayes

- Bernoulli, Precision score: 0.9047619047619048
- Bernoulli, Recall score: 0.3825503355704698

Despite having a relatively high accuracy the Bernoulli's classifier has a low recall and so misclassifes spams as easy-hams(false negative = 92).

3.3.2 Spam versus hard-ham

3.3.2.1 Multinomial Naive Bayes Classifier

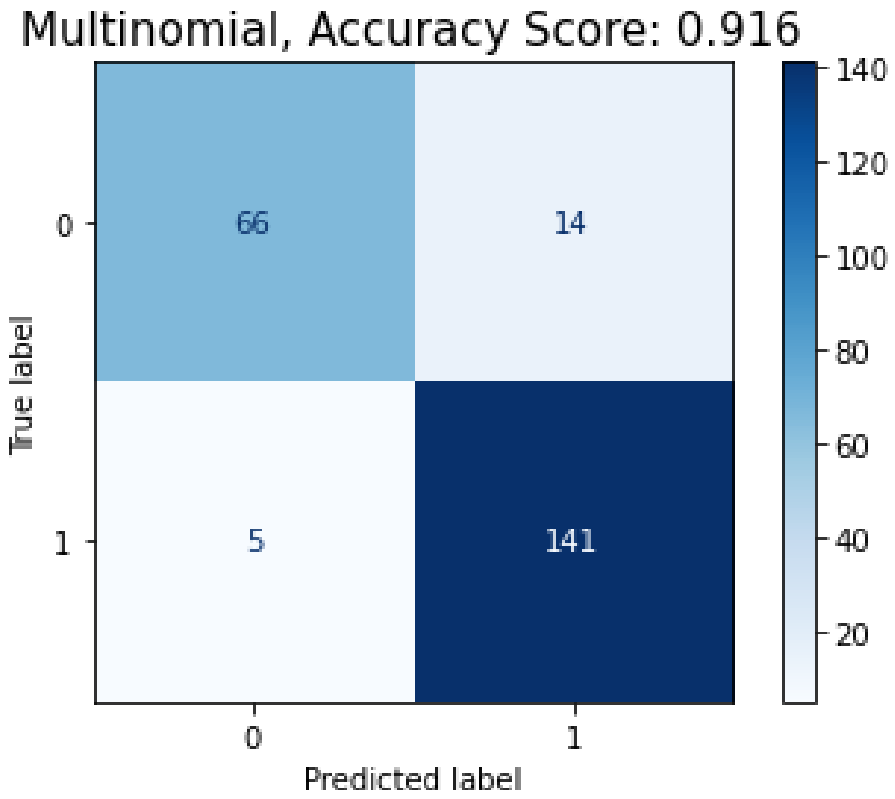

Figure 3: Spam versus hard-ham on Multinomial Naive Bayes

- Multinomial, Precision score: 0.9096774193548387
- Multinomial, Recall score: 0.9657534246575342

Since we have lower precision it indicates more false positives which means hard-ham are classified as spams and in our case this happened 14-times. It has comparatively high recall.

3.3.2.2 Bernoulli Naive Bayes Classifier

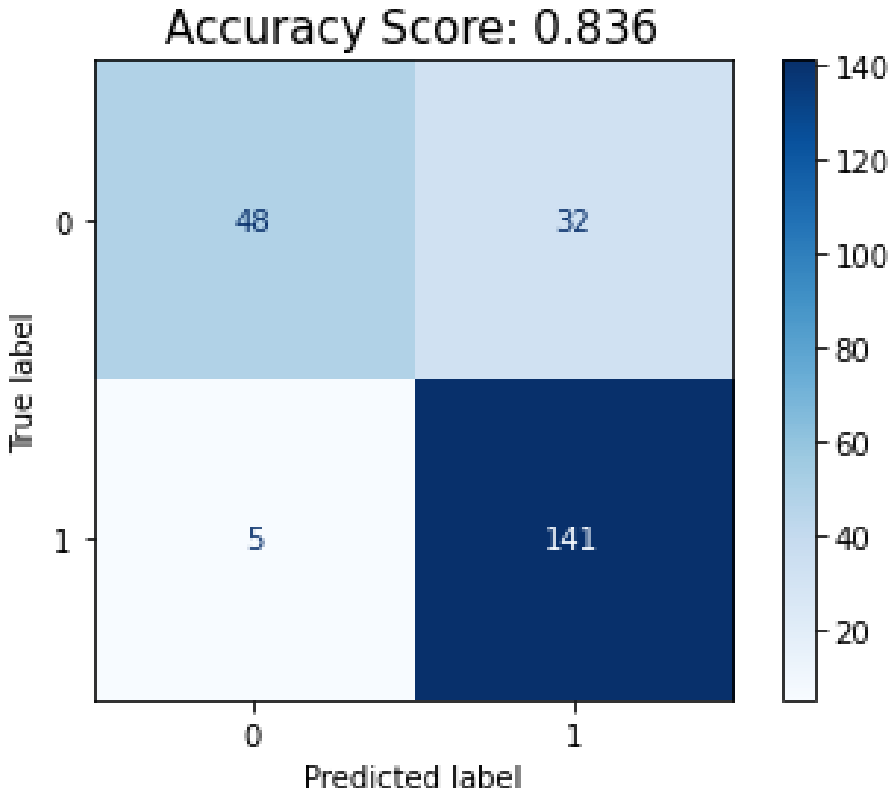

Figure 4: Spam versus hard-ham on Bernoulli Naive Bayes

- Bernoulli, Precision score: 0.815028901734104
- Bernoulli, Recall score: 0.9657534246575342

The Bernoulli's classifier for hard-hams has a lower precission score meaning it has more false postives and in this case missclassified 32 emails as spams instead of hard-hams.

3.4 Common and uninformative words

3.4.1 Finding the words that are too common/uncommon

In this approach was to find the word which appear frequently in the data set, since if these words too common it might not add much information to the text and especially would not give any key information for classifier.

Most common list words			
Easy-ham & Spam — Hard-ham & Spam			
Common word	Count	Common word	Count
the	25316	the	13653
to	16101	to	9572
of	13020	and	7415
and	12793	of	7183
a	11387	a	6176
in	8259	in	4753
is	7302	you	4285

Table 1: List of the most common words in the two data sets

In the Table 1, as we can see the most common word in both our data sets are belonging to the English **stop-words** which are low-level information from the text which do not give us the values we want for the classification.

Beside that, we also found two list of the uncommon word of two data sets. In the top of least common words, it mostly contain some link of websites, some html text formatting, emails, time-day. We also noticed that in the top list of least common words, those word only occur one or two times.

In conclusion, the words that are too common and too uncommon might not give the information to for our models in order to classifier the spam emails.

3.4.2 Filtering out unwanted words

In this part, we had two options to filter out the unwanted words in the data which are using Sklearn's CountVectorizer or reusing the word we found in the previous part. We chose using CountVectorizer for removing word that appear too frequently and the opposite. In particular, we set max_df for **0.90** which means we taking out the words that appear in more than 90% of the data, and min_df = **2** for ignoring the words that appear in less than 2 emails. Moreover, we also used **stop-words** in English language to filtered out the most common words in English which does not add high-values information to the text.

Consequently, we added the parameters we discussed above into our function in Task 3.2 and run it on spam versus easy-ham and spam versus hard-ham.

3.4.3 Spam versus easy-ham

3.4.3.1 Multinomial Naive Bayes Classifier

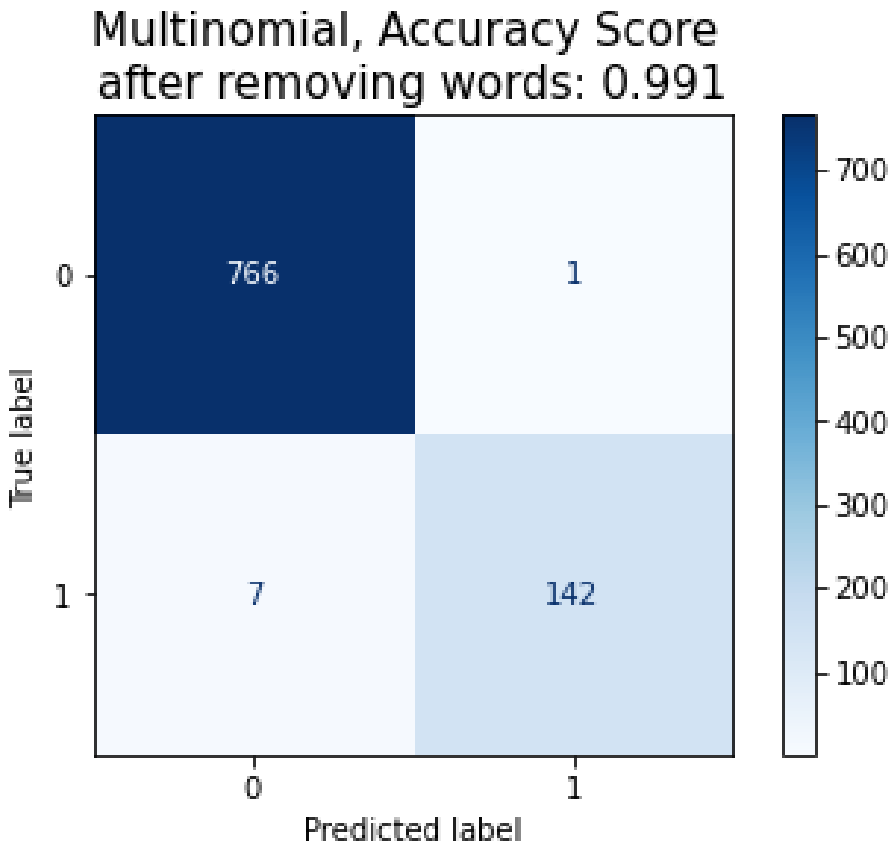

Figure 5: Spam versus easy-ham on Multinomial Naive Bayes

- Multinomial, Precision score: 0.993006993006993
- Multinomial, Recall score: 0.9530201342281879

Multinomial classifier has a high precission score but a lower recall, so it tends to misclassify spams as easy-hams(false negative = 7).

3.4.3.2 Bernoulli Naive Bayes Classifier

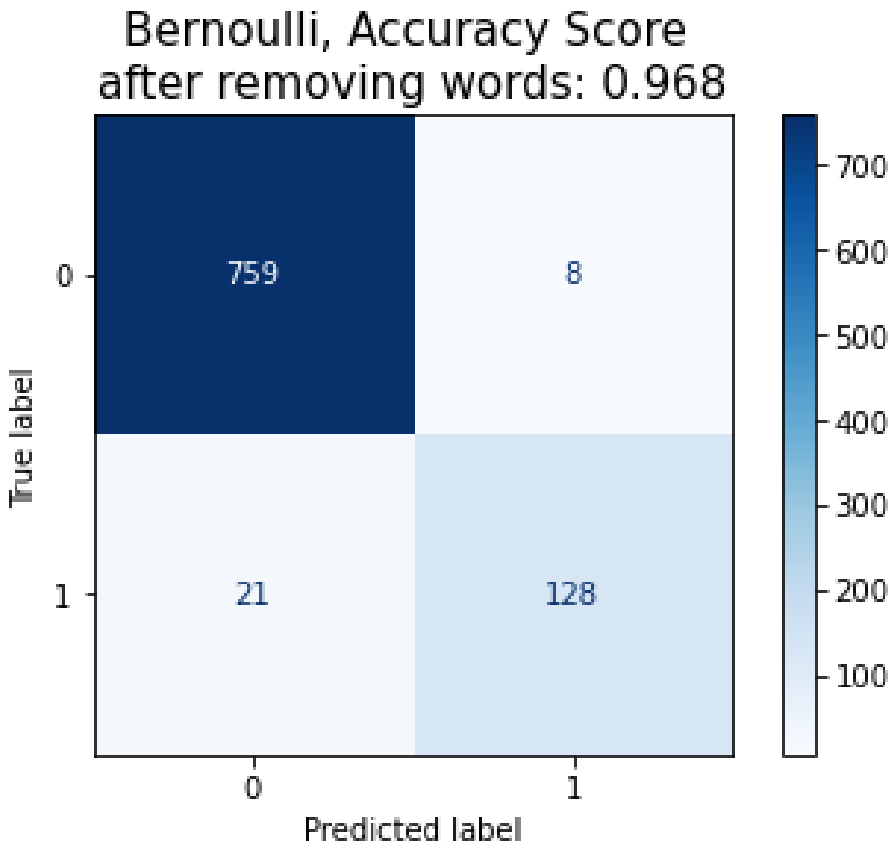

Figure 6: Spam versus easy-ham on Bernoulli Naive Bayes

- Bernoulli, Precision score: 0.9411764705882353
- Bernoulli, Recall score: 0.8590604026845637

Bernoulli classifier has a lower recall score, so it misclassifies spams as hard-hams(false negative = 21).

3.4.4 Spam versus hard-ham

3.4.4.1 Multinomial Naive Bayes Classifier

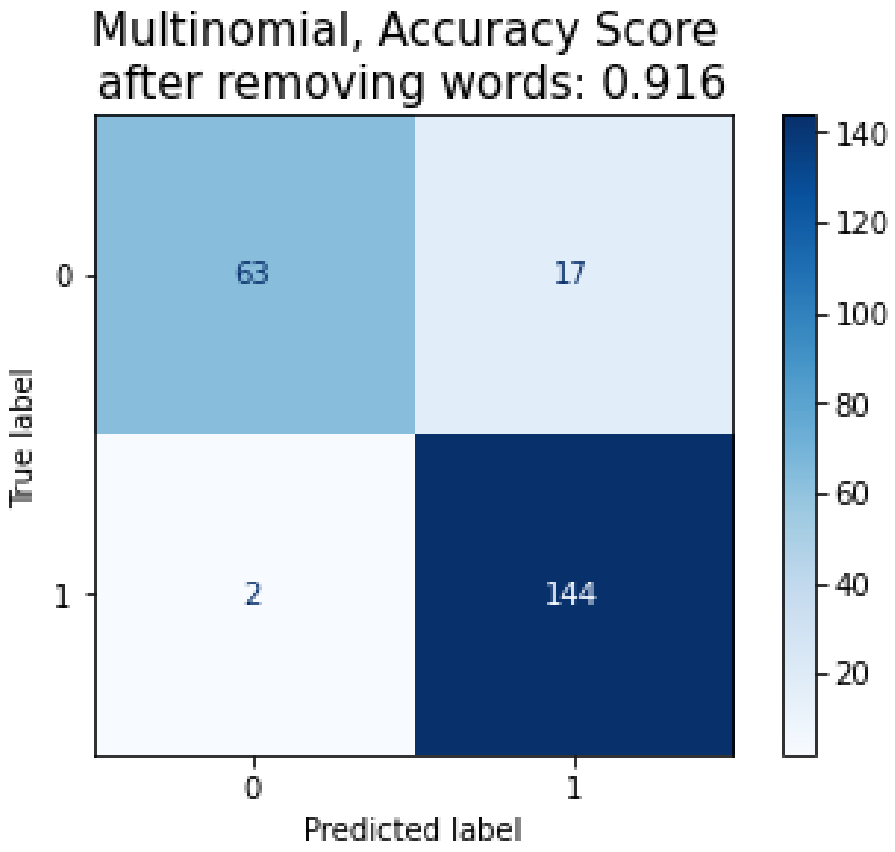

Figure 7: Spam versus hard-ham on Multinomial Naive Bayes

- Multinomial, Precision score: 0.8944099378881988
- Multinomial, Recall score: 0.9863013698630136

Multinomial classifier has a lower precision score and misclassifies hard-hams as spams(false positive = 17).

3.4.4.2 Bernoulli Naive Bayes Classifier

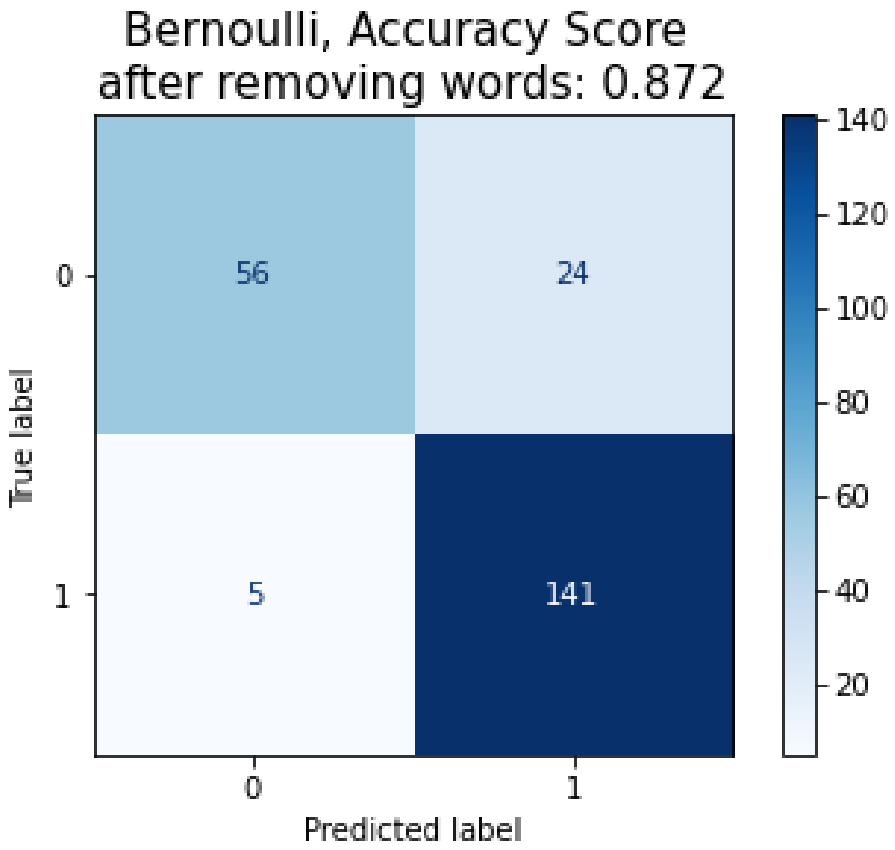

Figure 8: Spam versus hard-ham on Bernoulli Naive Bayes

- Bernoulli, Precision score: 0.8545454545454545
- Bernoulli, Recall score: 0.9657534246575342

Bernoulli classifier has a higher recall and missclasifies hard-hams as spams(false positive = 24).
Overall removing the common words has increased the accuracy for both multinomial and Bernoulli classifiers when classifying easy-ham and spam, and hard-ham and spam.

3.5 Filter out the headers and the footers.

In a few emails, we noticed that the header footer usually started with some string, so we decided to remove these lines. Examples of a few words at the

start of the header and footer of an email: "From", "Return", "Received", "Delivered", "Sender", "HTTP".

3.5.1 Spam versus easy-ham

3.5.1.1 Multinomial Naive Bayes Classifier

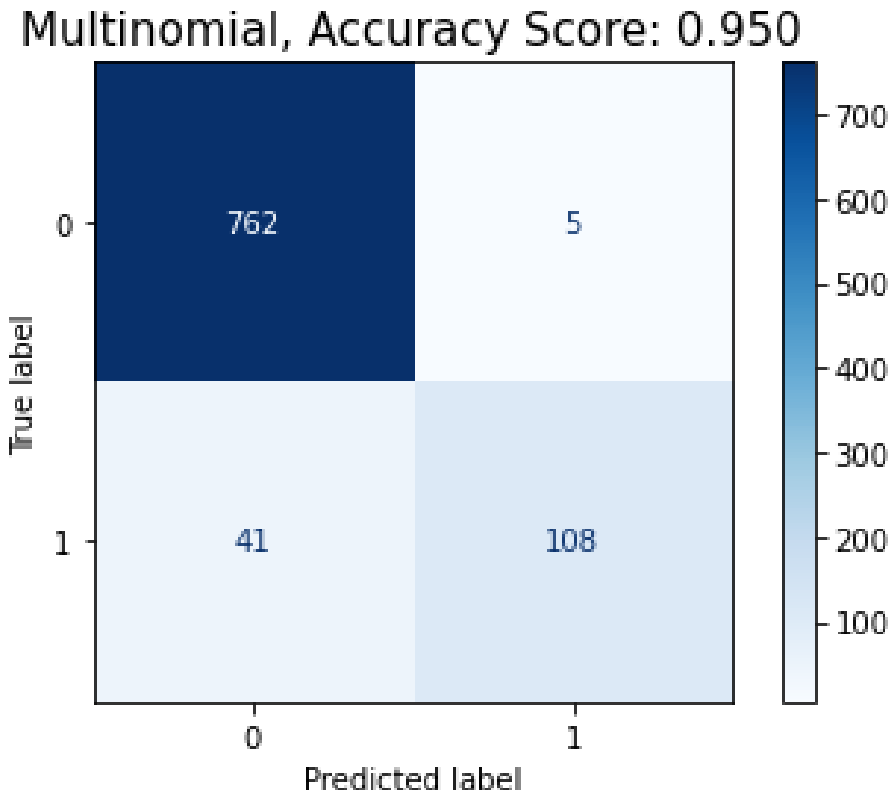

Figure 9: Spam versus easy-ham on Multinomial Naive Bayes

- Multinomial, Precision score: 0.9557522123893806
- Multinomial, Recall score: 0.7248322147651006

Multinomial clasifier has a higher precision score and lower recall hence misclasifies spams as easy-hams(false negative = 41).

3.5.1.2 Bernoulli Naive Bayes Classifier

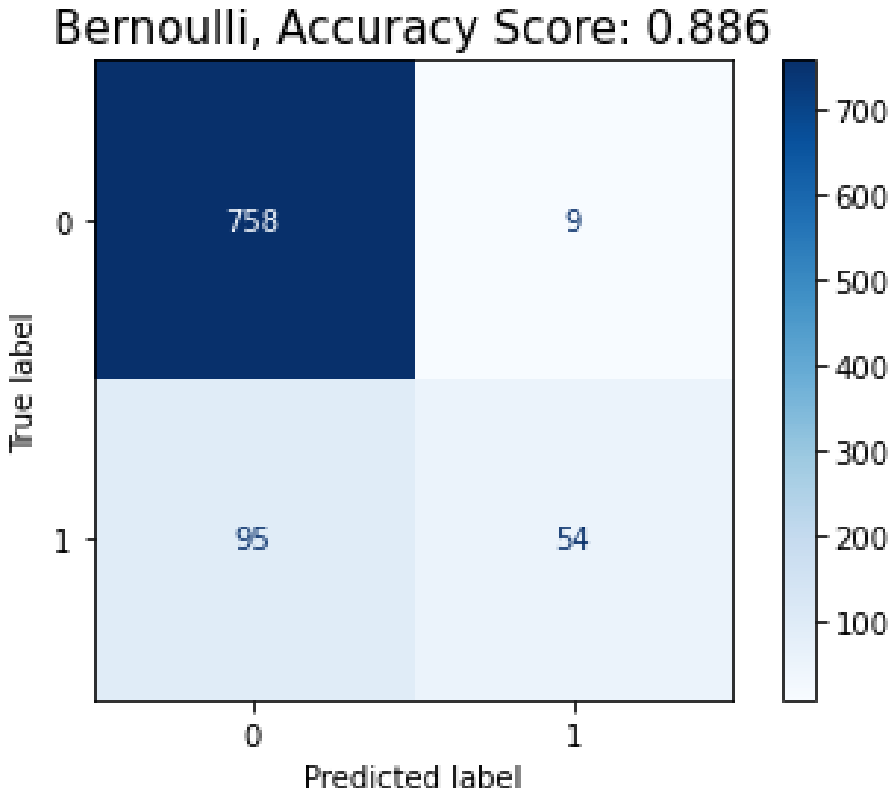

Figure 10: Spam versus easy-ham on Bernoulli Naive Bayes

- Bernoulli, Precision score: 0.8571428571428571
- Bernoulli, Recall score: 0.3624161073825503

Bernoulli classifier has a higher precision score and a lower recall score hence misclassifies spam as easy-ham(false negative = 95).

3.5.2 Spam versus hard-ham

3.5.2.1 Multinomial Naive Bayes Classifier

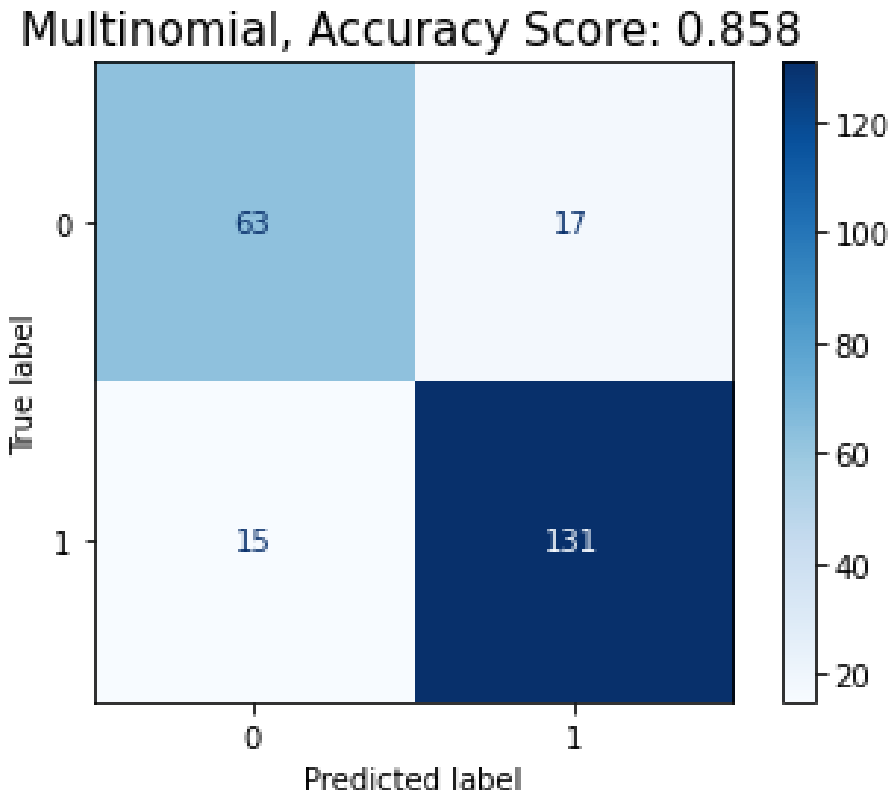

Figure 11: Spam versus hard-ham on Multinomial Naive Bayes

- Multinomial, Precision score: 0.8851351351351351
- Multinomial, Recall score: 0.8972602739726028

Multinomial classifier has lower precision score and higher recall, so misclassifies hard-hams as spams(false positive = 17).

3.5.2.2 Bernoulli Naive Bayes Classifier

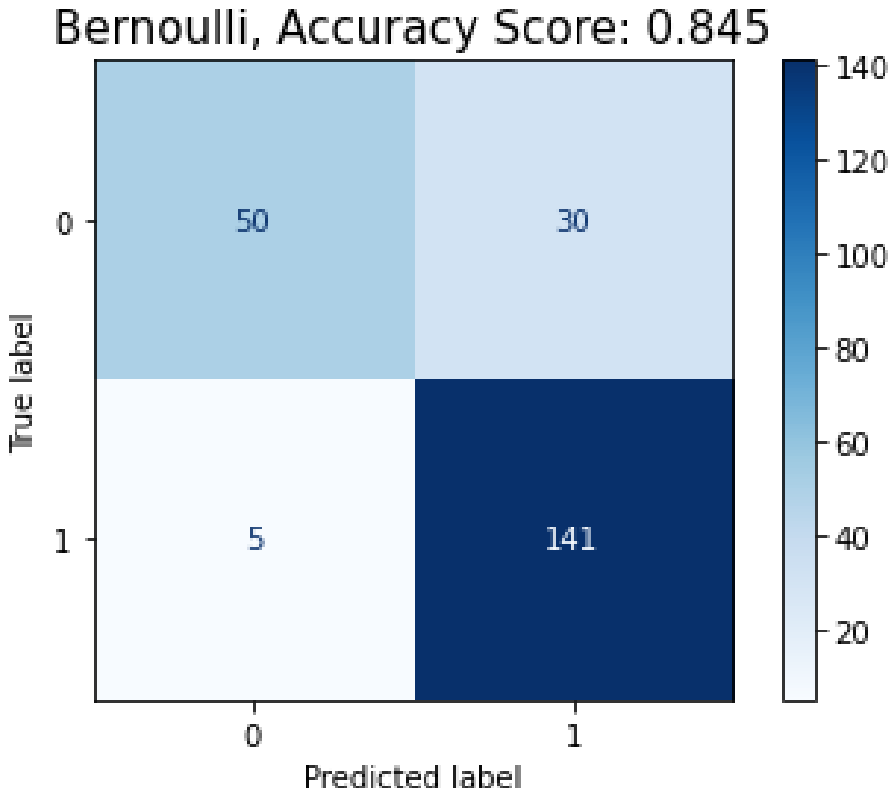

Figure 12: Spam versus hard-ham on Bernoulli Naive Bayes

- Bernoulli, Precision score: 0.8245614035087719
- Bernoulli, Recall score: 0.9657534246575342

Bernoulli classifier has a higher recall and a lower precision score, so misclassifies hard-ham with spam(false positive = 30).

3.5.3 Spam versus easy-ham

3.5.3.1 Multinomial Naive Bayes Classifier

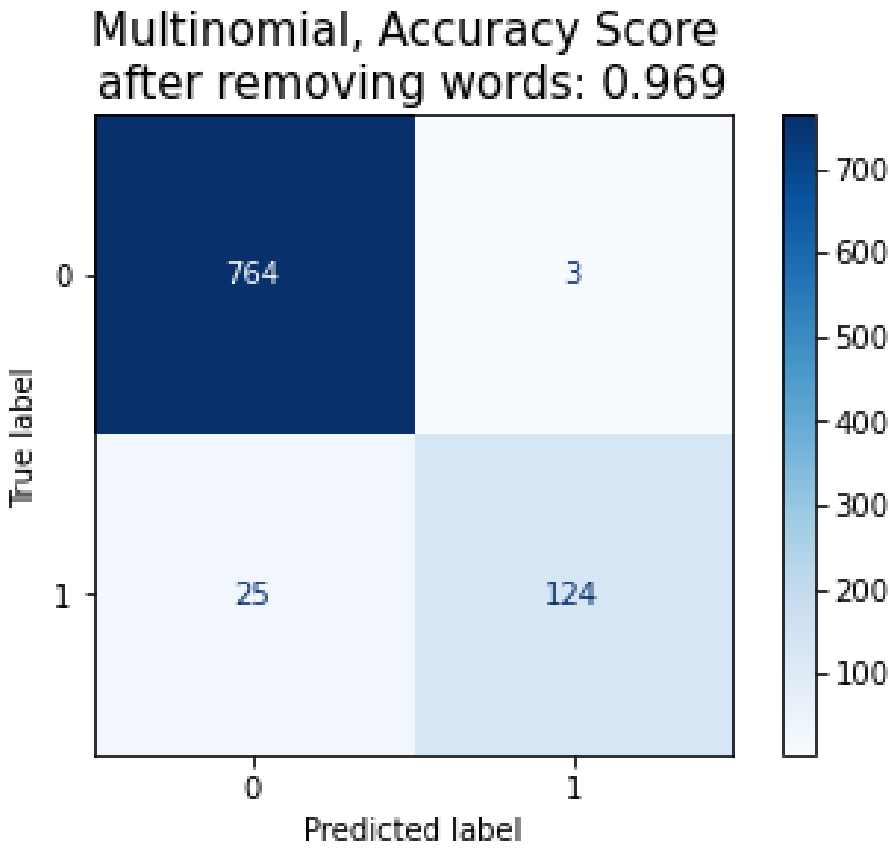

Figure 13: Spam versus easy-ham on Multinomial Naive Bayes

- Multinomial, Precision score: 0.9763779527559056
- Multinomial, Recall score: 0.8322147651006712

Multinomial classifier has a higher precision and a lower recall score, so misclassifies spam as easy-ham(false negative = 25).

3.5.3.2 Bernoulli Naive Bayes Classifier

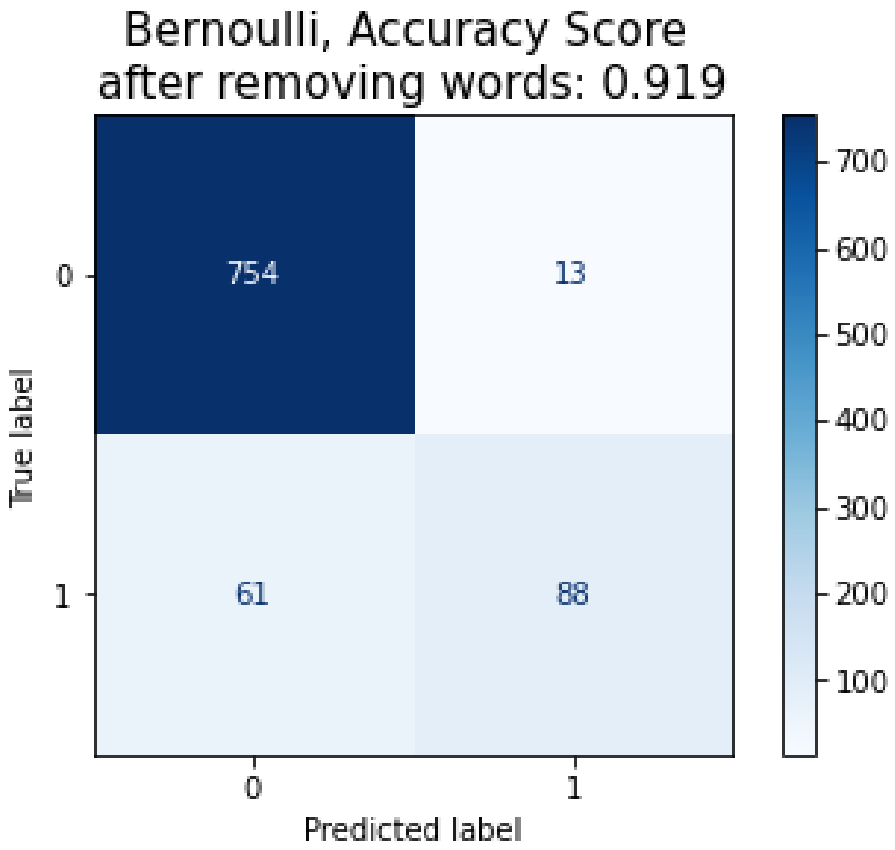

Figure 14: Spam versus easy-ham on Bernoulli Naive Bayes

- Bernoulli, Precision score: 0.8712871287128713
- Bernoulli, Recall score: 0.5906040268456376

Bernoulli classifier has a higher precision and a lower recall score, so misclassifies spam as easy-ham(false negative = 61).

3.5.4 Spam versus hard-ham

3.5.4.1 Multinomial Naive Bayes Classifier

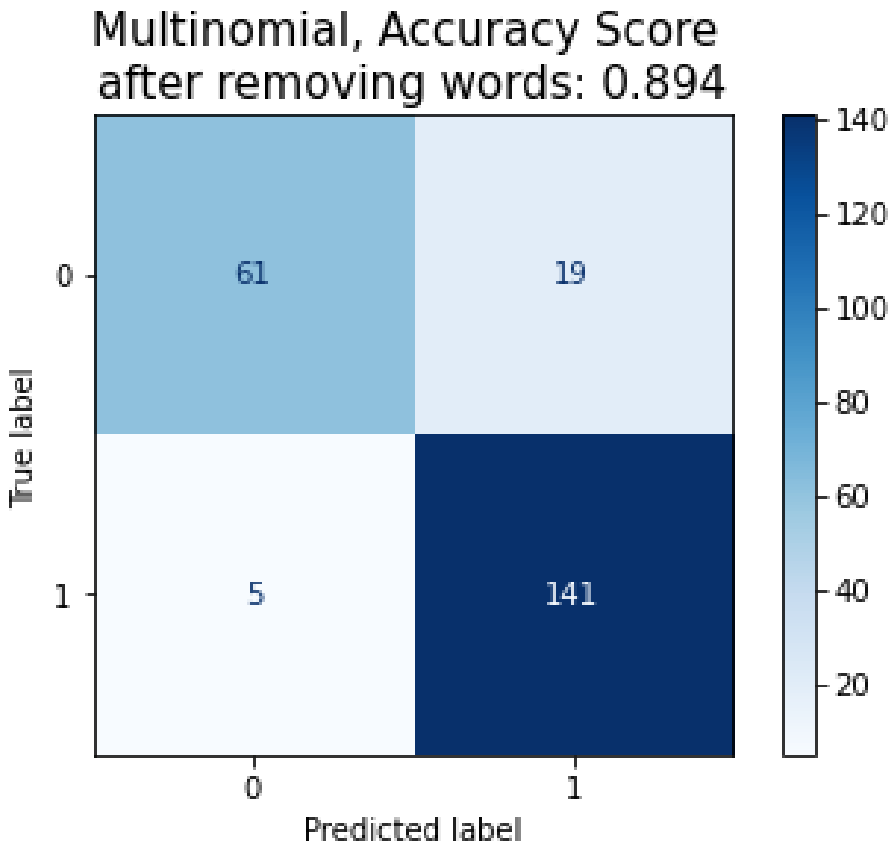

Figure 15: Spam versus hard-ham on Multinomial Naive Bayes

- Multinomial, Precision score: 0.88125
- Multinomial, Recall score: 0.9657534246575342

Multinomial classifier has a higher recall and a lower precison score, so misclassifies hard-hams as spams(false positive = 19).

3.5.4.2 Bernoulli Naive Bayes Classifier

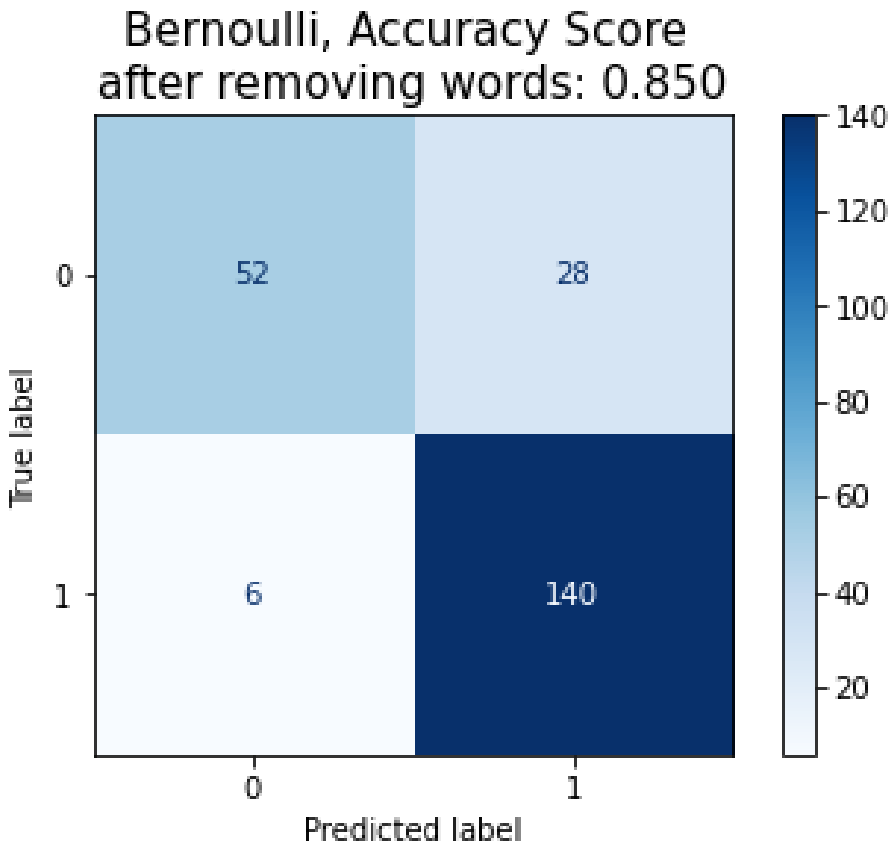

Figure 16: Spam versus hard-ham on Bernoulli Naive Bayes

- Bernoulli, Precision score: 0.8333333333333334
- Bernoulli, Recall score: 0.958904109589041

Bernoulli classifier has a higher recall and a lower precison score, so misclassifies hard-ham as spam(false positive = 28).

Overall removing the header and footer from the emails has reduced the accuracy for both multinomial and Bernoulli classifiers when classifying easy-ham and spam, and hard-ham and spam.

From the Table 2, the best accuracy 99.1% is when we use the original data without filter out the header and the footer but remove the most frequent words

Ham	Classifier Type	Accuracy	Precision	Recall
Easy	Multinomial task 3	0.966	0.9916	0.7986
Hard	Multinomial task 3	0.916	0.9096	0.9675
Easy	Bernoulli task 3	0.893	0.9047	0.3825
Hard	Bernoulli task 3	0.836	0.8150	0.9657
Easy	Multinomial task 4	0.991	0.9930	0.9530
Easy	Bernouli task 4	0.968	0.9411	0.8590
Hard	Multinomial task 4	0.916	0.8944	0.9863
Hard	Bernoulli task 4	0.872	0.8545	0.9657
Easy	Multinomial task 53	0.950	0.9557	0.7248
Easy	Bernoulli task 53	0.886	0.8571	0.3624
Hard	Multinomial task 53	0.858	0.8851	0.8972
Hard	Bernoulli task 53	0.846	0.82456	0.96567
Easy	Multinomial task 54	0.969	0.97637	0.83221
Easy	Bernoulli task 54	0.919	0.8712	0.59060
Hard	Multinomial task 54	0.894	0.88125	0.96575
Hard	Bernoulli task 54	0.850	0.8333	0.95898

Table 2: Table on accuracy, precision and Recall of the columns in the dataset.

with appear more than 90% and least common word which appear less than 2 emails in the data. So we can conclude that when we filtered out the header and footer we may remove some values information that lead to lower the accuracy of the model.

3.5.5 Skewed results while splitting the data set

As we mentioned in Task 3.1 our data set is imbalance between the number of spam emails and non-spam emails. This imbalance issue may have affected on the model's performance. Particularly, in the easy-spam data, we only have 20% of the dataset is spam emails, and we split the model with 30% is test and 70 were used to train the models. Therefore the percentage of spam emails in the training set is deficient. Since the distribution of the training set is highly different between spam and non-spam, the models might not have much information for spam to classifier with the non-spam emails.

The solution for this case is we need to consider the data set ratio and make sure there the balancing between the two class for binary classifier. We could use **stratified sampling** to ensure the distribution in the training set will be exactly the same.

3.5.6 If your training set were mostly spam messages while your test set were mostly ham messages?

If the there were mostly spam emails in the training set, the model would have not be able to get use to the non-spam emails and easy to tend to define a non-spam message as a spam message. And also this will increase the number of false positive (true non-spam emails, predicted as spam emails).