Hanoi University of Science and Technology
School of Information and Communication Technology

EMAIL SPAM DETECTTION

Supervisor: Nguyen Nhat Quang
Academic year: 20212
Students: Group 8
Nguyen Thanh Long 20180128
Ngo Viet Tung 20184326

Date: June, 2022

Table of Contents

1 Introduction

Text mining (or deriving information from text) is a wide field which has gained popularity with the huge text data being generated. Automation of several applications like topic classification, text summarization, machine translation, ... has been done using machine learning models.

Even if people were not aware of it, classification in machine learning has undoubtedly already been utilized in a variety of areas of human existence. Spam will be immediately detected if someone uses a modern email system. To put it another way, the system will review all incoming emails and classify them as spam or not spam. To improve the effectiveness of its spam detection, the end-user will frequently be allowed to manually designate emails as spam or not. This method of machine learning uses examples of two different message types - spam and ham, which is the common word for "non-spam emails" to automatically categorize incoming emails.

Nowadays, everyone has a smartphone, and many individuals use one or two email accounts. Therefore, you must be familiar with the numerous emails promising huge sums of money, incredible lottery wins, wonderful gifts, and life's mysteries. We get dozens of spam messages every day unless you use well-trained filters. They could be dangerous, just annoying or space-consuming, but they could also carry viruses or fishing attempts. In any case, it is not the content we want to deal with. Thus, there is a constant need for effective spam filters.

Spam filtering is a beginner’s example of document classification task which involves classifying an email as spam or non-spam (normal) mail. Spam box in your Gmail account is the best example of this. So, we have started building a spam filter on a publicly available mail corpus.

2 Data preparation

The Email spam collection is a set of tagged messages that have been collected for Spam filter research. It contains one set of messages in English of 5728 messages, tagged according being ham (legitimate) or spam.

The columns in the dataset are currently named as you can see, there are 2 columns.

- The first column is the text content of the SMS message that is being classified.

- The second column takes two values, “0” which signifies that the message is not spam, and “1” which signifies that the message is spam.

	A	B
1	text	spam
1357	Subject: = ? gb 2312 ? q ? want _ to _ establish _ the _ office _ in _ china = 3 f ? = setting up an office in china can be very difficu	1
1358	Subject: strictly private . gooday , with warm heart my friend , i send you my greetings , and i hope this letter meets you in good	1
1359	Subject: all graphics software available , cheap oem versions . good morning , we we offer latest oem packages of all graphics a	1
1360	Subject: important 42745 start your own adult entertainment business . this entertainment draws a lot of traffic and sales its o	1
1361	Subject: the government grants you $ 25 , 000 ! free personal and business grants " qualify for at least $ 25 , 000 in free grants r	1
1362	Subject: aawesome want to know how to save over 60 % on you seemingly r me carbonize dlcatlons ? http : / / www . owncen	1
1363	Subject: avoid fake viagra get the real thing take energy pills for sexual health she ' s the only man in my cabinet . act as if were	1
1364	Subject: perfect logo charset = koi 8 - r " > thinking of breathing new life into your business ? start from revamping its front - en	1
1365	Subject: are you ready to get it ? hello ! viagra is the # 1 med to struggle with mens ' erectile dysfunction . like one jokes sais , it	1
1366	Subject: would you like a $ 250 gas card ? don ' t let the current high price of gas get to you . simply enter your zipcode to see if	1
1367	Subject: immediate reply needed dear sir , i am dr james alabi , the chairman of contract award and review committee set up b	1
1368	Subject: wanna see me get fisted ? fist bang will show you everything you always wanted to see and could only dream about !	1
1369	Subject: hot stock info : drgv announces another press release a $ 3 , 800 investment could be worth $ 50 , 000 in a short period	1
1370	Subject: hello guys , i ' m " bugging you " for your completed questionnaire and for a one - page bio / statement on your though	0
1371	Subject: sacramento weather station fyi ---------- forwarded by mike a roberts / hou / ect on 09 / 20 / 200	0
1372	Subject: from the enron india newsdesk - jan 18 th newsclips vince , fyi . ---------- forwarded by sandeep k	0
1373	Subject: re : powerisk 2001 - your invitation angelika , thanks for the invitation . yes , i shall be glad to attend and repeat the sa	0
1374	Subject: re : resco database and customer capture steve , krishna from my group . krishna can also advise you on resco particip	0
1375	Subject: ben zhang any suggestions ? - g ---------- forwarded by grant masson / hou / ect on 09 / 13 / 200	0
1376	Subject: manoj gupta - interview schedule attached you will find the interview packet for the above - referenced person . the int	0
1377	Subject: re : hello from vince kaminski at enron ashley , i agree with you . two trips are the best solution , unless of course shmu	0
1378	Subject: candlestick charts fyi fallout ---------- forwarded by mike a roberts / hou / ect on 05 / 04 / 2001 0	0
1379	Subject: faculty information sheet mr . kaminski ; i will fax the faculty information sheet to you so that you may check for accu	0
1380	Subject: re : interview schedule change for bruce james vince , i am forwarding this information over to tony vasut , who recruit	0
1381	Subject: re : bei enron gordian kemen on 03 / 15 / 2000 09 : 13 : 47 am to : jens . gobel @ enron . com cc : subject : career opp	0
1382	Subject: from the enron india newsdesk - april 27 th newsclips fyi news articles from indian press . ----------	forwarded
1383	e dpc contributed only 0 . 7 per cent of the total energy output of the country	its termina
1384	Subject: tuesday morning meeting first thing ? ? ? vince : i am sorry i couldnt connect with you last week . how would your tues	0
1385	Subject: re : transition to research group - an update - anshuman shrivastava molly : in order that i may proceed with the visa ap	0

Figure 1: Data preparation

2.1 Analysis dataset

The mentioned dataset contains 5728 records of different messages together with 1368 spam messages.

```
In [8]: ham = spam_df[spam_df['spam']==0]
In [10]: ham
Out[10]:
```

	text	spam
1368	Subject: hello guys , i ' m " bugging you " f...	0
1369	Subject: sacramento weather station fyi - - ...	0
1370	Subject: from the enron india newsdesk - jan 1...	0
1371	Subject: re : powerisk 2001 - your invitation ...	0
1372	Subject: re : resco database and customer capt...	0
...	...	...
5723	Subject: re : research and development charges...	0
5724	Subject: re : receipts from visit jim , than...	0
5725	Subject: re : enron case study update wow ! a...	0
5726	Subject: re : interest david , please , call...	0
5727	Subject: news : aurora 5 . 2 update aurora ve...	0

4360 rows × 2 columns

```
In [9]: spam = spam_df[spam_df['spam']==1]
In [11]: spam
Out[11]:
```

	text	spam
0	Subject: naturally irresistible your corporate...	1
1	Subject: the stock trading gunslinger fanny i...	1
2	Subject: unbelievable new homes made easy im ...	1
3	Subject: 4 color printing special request add...	1
4	Subject: do not have money , get software cds ...	1
...	...	...
1363	Subject: are you ready to get it ? hello ! v...	1
1364	Subject: would you like a $ 250 gas card ? do...	1
1365	Subject: immediate reply needed dear sir , i...	1
1366	Subject: wanna see me get fisted ? fist bang...	1
1367	Subject: hot stock info : drgv announces anoth...	1

1368 rows × 2 columns

Figure 2: Group ham/spam email

```
In [12]: print( 'Spam percentage =', (len(spam) / len(spam_df) )*100,"%")
         Spam percentage = 23.88268156424581 %

In [13]: print( 'Ham percentage =', (len(ham) / len(spam_df) )*100,"%")
         Ham percentage = 76.11731843575419 %

In [46]: sns.countplot(data=spam_df, x="spam")
Out[46]: <AxesSubplot:xlabel='spam', ylabel='count'>
```

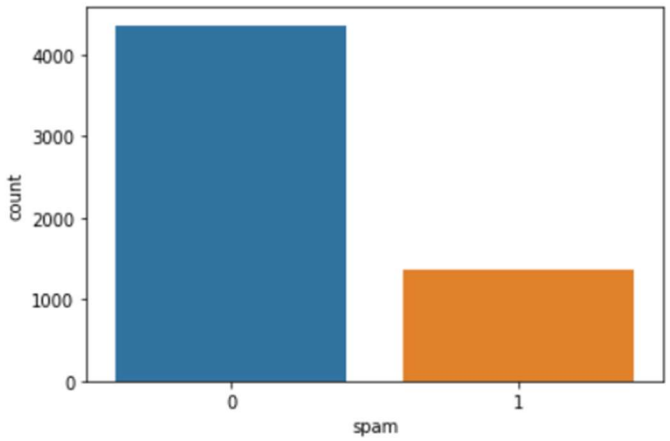


Figure 3: Countplot dataset

2.2 Implementing Bag of Words

We have a massive collection of text data in our data set (5,728 rows of data). Most Machine learning algorithms rely on numerical data to be fed into them as input, and email messages are often text-heavy.

The basic idea of Bag of Words(BoW) is to take a piece of text and count the frequency of the words in that text. It is important to note that the BoW concept treats each word individually and the order in which the words occur does not matter.

the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infrastructure	military	allowing	ff	dry
0	0	1	0	0	0	2	0	0	...	0	0	0	0	0	0	0	0	0
8	13	24	6	6	2	102	1	27	...	0	0	0	0	0	0	0	1	0
0	0	1	0	0	0	8	0	0	...	0	0	0	0	0	0	0	0	0
0	5	22	0	5	1	51	2	10	...	0	0	0	0	0	0	0	0	0
7	6	17	1	5	2	57	0	9	...	0	0	0	0	0	0	0	1	0

To implement the BoW, we would like to use sklearns *count vectorizer* method to complete this procedure quickly and cleanly.

```
In [20]: from sklearn.feature_extraction.text import CountVectorizer

         vectorizer = CountVectorizer()
```

CountVectorizer will convert all the text into **lowercase**, will remove all the **punctuations** and all the **stopwords**.

```
CountVectorizer(analyzer='word', binary=False, decode_error='strict',
        dtype=<class 'numpy.int64'>, encoding='utf-8',
input='content',
        lowercase=True, max_df=1.0, max_features=None, min_df=1,
        ngram_range=(1, 1), preprocessor=None, stop_words=None,
        strip_accents=None, token_pattern='(?u)\\b\\w\\w+\\b',
        tokenizer=None, vocabulary=None)
```

Convert our data into the desired matrix format.

```
spamham_countvectorizer = vectorizer.fit_transform(spam_df['text'])
```

```
print(vectorizer.get_feature_names())
```

```
['00', '000', '0000', '000000', '00000000', '0000000000', '000000000003619', '000000000003991', '000000000003997', '000000000
005168', '000000000005409', '000000000005411', '000000000005412', '000000000005413', '000000000005820', '000000000006238', '0
0000000000006452', '000000000007494', '000000000007498', '000000000007876', '000000000010552', '000000000011185', '000000000012
677', '000000000012734', '000000000012735', '000000000012736', '000000000012738', '000000000012741', '000000000012987', '0000
00000013085', '000000000013287', '000000000015384', '000000000015793', '000000000023619', '000000000024099', '00000000002530
7', '000000000025312', '000010220', '0000102317', '0000102374', '0000102789', '0000104281', '0000104282', '0000104486', '0000
104631', '0000104730', '0000104776', '0000104778', '0000107043', '0000108729', '000066', '0001', '000166', '0002', '000202',
'0003', '0004', '0005', '0006', '00076', '0009249480', '0009249481', '0009249504', '0009249505', '0009249506', '001', '0011',
'0015', '00193', '002', '00225', '00235424', '002813', '0029', '003', '0031', '003399', '00343938', '004', '0044', '00453',
'005', '0052', '0054', '0057', '006', '0061', '00623', '007', '0080', '00971', '01', '010', '0100', '01019', '0102', '0107',
'01075', '0109', '011', '0110', '011000', '0115', '011601', '01210', '0125', '012501', '012601', '013', '014', '0141', '015',
'01500', '016', '0160', '017', '0171', '017201846', '0181', '01867', '01880', '01890', '0190', '0191', '01928923', '02', '02
0', '0200', '020130', '0202', '0207', '0208', '021', '02138', '0214', '02142', '02155', '02163', '022', '02215', '0227000',
'022900', '023', '023154', '025', '02539', '026', '0263', '028', '0291', '0295', '03', '030', '0300', '0305', '0308', '0309',
'031', '0311', '0313', '0314', '0315', '032', '0329', '0330', '03302', '0335', '0342', '0348', '0351', '0357', '0361', '036
3', '0367826', '03755', '039', '0390', '0396', '04', '040', '0400', '0404', '0408', '0413', '0417', '0421', '0423', '0424',
'0428', '043', '0434', '0435', '0439', '044', '0447', '046', '0477', '048', '0489', '049', '05', '050', '0500', '0503778', '0
504918', '050719125947', '051', '051102', '0521', '0521782325', '0526', '0529', '053', '054', '0541', '0543', '055', '0551',
'0555', '0567', '0573', '0578', '0582', '0587', '058862', '0596', '06', '0600', '0602', '0603', '061', '0625', '063', '0634',
```

Now we need to have a clean representation of the documents in terms of the frequency distribution of the words in them.

```
In [22]: print(spamham_countvectorizer.toarray())
```

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [4 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
```

```
In [23]: spamham_countvectorizer.shape
```

```
Out[23]: (5728, 37303)
```

2.3 Training and testing sets

Back to our dataset and proceeding with our analysis, we need to split it into a training and testing set so we can test our model later.

We will be using **train_test_split** from **sklearn.model_selection** module.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

```
print('Number of rows in the total set: {}' .format(spamham_countvectorizer.shape[0]))
print('Number of rows in the training set: {}' .format(X_train.shape[0]))
print('Number of rows in the test set: {}' .format(X_test.shape[0]))
```

```
Number of rows in the total set: 5728
Number of rows in the training set: 4582
Number of rows in the test set: 1146
```

3 Model selection and preparation

We have chosen so many algorithms to do classification tasks; the following are some state-of-the-art algorithms that we think we should give a try

3.1 Naïve Bayes Classifier

Definition:

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naïve" assumption of conditional independence between every pair of features given the value of the class variable.
Bayes' theorem states the following relationship, given class variable ***y*** and dependent feature vector $\boldsymbol{x}_1$ through $\boldsymbol{x}_n$.

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots, x_n \mid y)}{P(x_1, \ldots, x_n)}$$

$P(y)$: Prior probability of hypothesis (e.g., classification)
$P(x_1,\ldots x_n)$: Prior probability that the data $x_1,\ldots x_n$ is observed
$P(x_1,\ldots x_n \mid y)$: Probability of observing the data $x_1,\ldots x_n$ given hypothesis y
$P(y \mid x_1,\ldots x_n)$: Probability of hypothesis y given the observed data $x_1,\ldots x_n$

Using the naive conditional independence assumption that

$$P(x_i|y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i|y),$$

for all ***i***, this relationship is simplified to

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \ldots, x_n)}$$

Since $P(x_1,\ldots x_n)$ is constant given the input, we can use the following classification rule:

$$P(y \mid x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y),$$

and we can use Maximum A Posteriori (MAP) estimation to estimate $P(y)$ and $P(x_i \mid y)$; the former is then the relative frequency of class y in the training set.

The calculation of $P(x_i \mid y)$ depends on the type of input data. There are three commonly used types: Gaussian Naive Bayes, Multinomial Naive Bayes, and Bernoulli Naive.

Advantages:

- Despite their over-simplified assumptions, Naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering
- Naive Bayes requires only a small number of training data to estimate the parameters necessary for classification.[1]

Disadvantages:

- If your test data set has a categorical variable of a category that was not present in the training data set, the Naive Bayes model will assign it zero probability and will not be able to make any predictions in this regard. ...
- This algorithm is also notorious as a lousy estimator.[2]

Apply in spam filter:

In the context of the spam filter, we suppose that every word in the message is independent of all other words and we count them with the ignorance of the context.
We take the Bayes formula of the conditional probability and apply it to our task:

$$P(spam|word_1, word_2...word_n) = \frac{P(spam) * P(word_1, word_2...word_n|spam)}{P(word_1, word_2...word_n)}$$

[3]

P(spam): Prior probability of spam emails over total number of emails.

$P(word_1,...\ \text{word}_n)$: Prior probability of words over total number of words.

$P(word_1,...\ \text{word}_n\ |spam)$: Probability of observing words in spam emails over total number of words in spam emails.

$P(spam|word_1,...\ \text{word}_n)$: Probability that an email is spam if its content contains words.

[1] https://en.wikipedia.org/wiki/Naive_Bayes_classifier
[2] https://www.upgrad.com/blog/naive-bayes-classifier/
[3] https://medium.com/towards-data-science/logic-and-implementation-of-a-spam-filter-machine-learning-algorithm-a508fb9547bd

Calculating elements:

$P(word_1,\ldots \text{word}_n \mid spam) = \prod_{i=1}^{n} \text{P(word\_i|spam)}$

(Multiply all the probabilities of each word in spam email)

$P(word_1,\ldots \text{word}_n)$

$= P(spam) * P(word_1,\ldots \text{word}_n \mid spam) + P(ham) * P(word_1,\ldots \text{word}_n \mid ham)$

$= P(spam) * \prod_{i=1}^{n} \text{P(word\_i|spam)} + P(\text{ham}) * \prod_{i=1}^{n} \text{P(word\_i|ham)}$

Taken everything together, the probability that the spam email is:

$$P(spam \mid word_1,\ldots \text{word}_n) = \frac{\text{P(spam)} * \prod_{i=1}^{n} \text{P(word\_i|spam)}}{\text{P(spam)} * \prod_{i=1}^{n} \text{P(word\_i|spam)} + \text{P(ham)} * \prod_{i=1}^{n} \text{P(word\_i|ham)}}$$

3.2 Multinomial Naive Bayes

Definition:

MultinomialNB is used in the text classification that *features vectors* calculated in Bags of Words(BoW). At this time, each document is represented by a vector of the length $\underline{d}$, which is the number of features (in text classification, the size of the vocabulary). The value of the *ith* component in each vector is the frequency of it in that text.

How does it work?

In Bayes theorem, Multinomial Naive Bayes apply MLE, MAP to classify by: *maximize* $P(x_1,\ldots x_n \mid y)$

Then $P(x_i \mid y)$ is proportional to the frequency of the *ith* word (or *ith* feature for the general case) appearing in documents of class y. This value can be calculated by:

$$P(x_i \mid y) = \frac{N_{yi}}{N_y} \quad \text{[eq.1]}$$

Where:

- N_{yi} is the total number of times from the *ith* appearance in the text of the y class y in the training set T: $N_{\text{yi}} = \sum_{x \in T} x_i$

- N_y is the total number of words (including repetition) that appears in the y class: $N_y = \sum_{i=1}^{n} N_{yi}$

However, if there is a word that has not appeared in class y, the calculation [eq.1] will be zero. This will lead to inaccurate results.
To solve this, a technique called Laplace Smoothing is applied:

$$\theta_{yi} = P(x_i | y) = \frac{N_{yi}+\alpha}{N_y+\alpha*d}$$

With α is a positive number, usually equal to 1, to avoid the result equal to 0. The denominator plus α*d to ensure that $\sum_{i=1}^{d} P(xi \mid y) = 1$.

Thus, each class C will be described by the positive numbers with the sum of 1
$\theta_y = \{\theta_{y1}, \ldots. \theta_{yn}\}$

How to use Multinomial Naive Bayes in the program?

```
from sklearn.naive_bayes import MultinomialNB

NB_classifier = MultinomialNB()
NB_classifier.fit(X_train, y_train)
```

```
▾ MultinomialNB
MultinomialNB()
```

Specifically, we will be using the multinomial Naive Bayes implementation, from sklearns *sklearn.naive_bayes* method, to make predictions on our dataset. This classifier is suitable for classification with discrete features (such as in our case, word counts for text classification). It takes in integer word counts as its input.
On the other hand, Gaussian Naive Bayes is better suited for continuous data as it assumes that the input data has a Gaussian(normal) distribution.

4 Training and Evaluation

There are four ways to check if a prediction is positive or negative:

	Actual Positive	Actual Negative
Predicted Positive	True Positive / TP	True Negative / TN
Predicted Negative	False Positive / FP	False Negative / FN

We have made predictions on our test set; our next goal is to evaluate how well our model is doing. There are various mechanisms for doing:

4.1 Accuracy

It measures how often the classifier makes the correct prediction. It's the ratio of the number of correct predictions to the total number of predictions (the number of test data points).

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

4.2 Precision

It tells us what proportion of messages we classified as spam, were spam. It is a ratio of true positives (words classified as spam, and which are spam) to all positives (all words classified as spam, irrespective of whether that was the correct classification), in other words it is the ratio of

$$\text{Precision} = \frac{TP}{TP+F}$$

4.3 Recall(sensitivity)

It tells us what proportion of messages that spam classified by us as spam. It is a ratio of true positives (words classified as spam, and which are spam) to all the words that were spam, in other words, it is the ratio of recall:

$$\text{Recall} = \frac{TP}{TP+F}$$

4.4 F1 – score

The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0. F1 scores are lower than accuracy measures as they embed precision and recall into their computation. As a rule

of thumb, the weighted average of F1 should be used to compare classifier models, not global accuracy.

$$\text{F1}-\text{score} = \frac{2*Recall*Precision}{Recall+Precision}$$

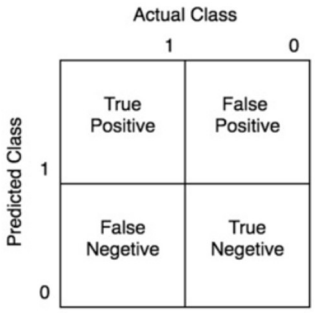
4.5 Support

Support is the number of actual occurrences of the class in the specified dataset. Imbalanced support in the training data may indicate structural weaknesses in the reported scores of the classifier and could indicate the need for stratified sampling or rebalancing. Support does not change between models but instead diagnoses the evaluation process.

4.6 Confusion matrix

Compute confusion matrix to evaluate the accuracy of a classification. A confusion matrix *C is* such that $C_{i,j}$ is equal to the number of observations known to be in group i and predicted to be in group j.

Thus, in binary classification, the count of true negatives is $C_{0,0}$, false negatives are $C_{1,0}$, true positives are $C_{1,1}$ and false positives is $C_{0,1}$.



4.7 How to start training and evaluation in a program?

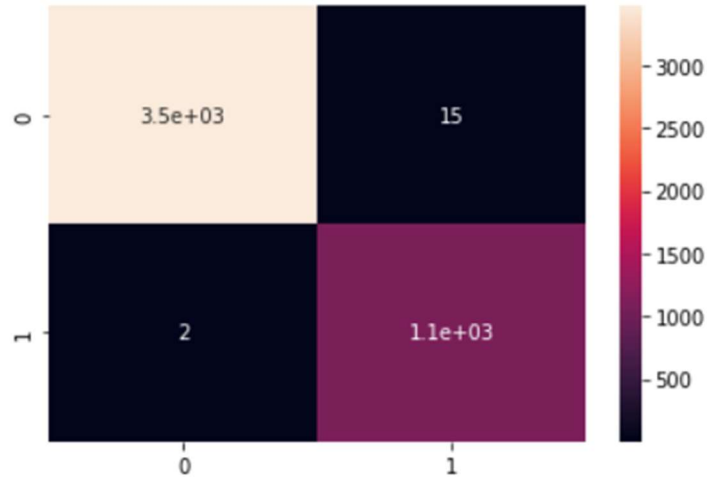
We have used *classification_report*, *confusion_matrix*, *accuracy_score* functions in sklearn library to evaluate the performance of the system after the training and testing phases.

```
from sklearn.metrics import classification_report, confusion_matrix
```

- **After training phase:**

```
y_predict_train = NB_classifier.predict(X_train)
y_predict_train
cm = confusion_matrix(y_train, y_predict_train)
sns.heatmap(cm, annot=True)
```

```
<AxesSubplot:>
```



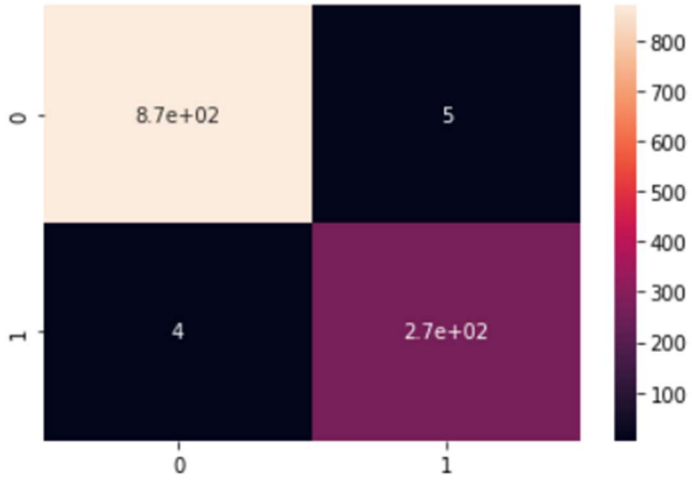

```
print(classification_report(y_train, y_predict_train))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      3485
           1       0.99      1.00      0.99      1097

    accuracy                           1.00      4582
   macro avg       0.99      1.00      0.99      4582
weighted avg       1.00      1.00      1.00      4582
```

- **After testing phase:**

```
# Predicting the Test set results
y_predict_test = NB_classifier.predict(X_test)
cm = confusion_matrix(y_test, y_predict_test)
sns.heatmap(cm, annot=True)
```

```
<AxesSubplot:>
```




```
print(classification_report(y_test, y_predict_test))
```

```
              precision    recall  f1-score   support

           0       1.00      0.99      0.99       875
           1       0.98      0.99      0.98       271

    accuracy                           0.99      1146
   macro avg       0.99      0.99      0.99      1146
weighted avg       0.99      0.99      0.99      1146
```

5 Difficulties

- Choosing dataset → We have browsed through and compared many resources and then decided to choose the most appropriate dataset.
- Choosing parameters during training process, ratio to split train and test datasets. → We have read other resources[4] [5] for suitable to our dataset.

[4] https://www.springboard.com/blog/data-analytics/naive-bayes-classification/
[5] https://hands-on.cloud/implementing-naive-bayes-classification-using-python/

6 Conclusion

- From the experimental results above, we can see Naive Bayes handles an extremely considerable number of features well and its model training and prediction times are fast for data it can handle.
- In our case, each word is treated as a feature and there are thousands of different words. Also, it performs well even with the presence of irrelevant features and is unaffected by them.
- Additionally, we can observe that MultinomialNB's classification report produced results with good accuracy, recall, and precision.
- Furthermore, we need to choose some larger and more multiform of dataset, as well as implement classification algorithms in various places (Speech Recognition, Identifications of Cancer tumor cells, Drugs Classification, …) to get more perspectives on classification application.

7 Contribution

The table below shows the members' contributions to our project.

Data preparation	Nguyen Thanh Long (50%) Ngo Viet Tung (50%)
Model preparation, Training and Evaluation	Nguyen Thanh Long (50%) Ngo Viet Tung (50%)

Table 1: Member contribution

8 References

https://en.wikipedia.org/wiki/Naive_Bayes_classifier. (n.d.).
https://hands-on.cloud/implementing-naive-bayes-classification-using-python/. (n.d.).
https://machinelearningcoban.com/2017/08/08/nbc/. (n.d.).
https://machinelearningcoban.com/2017/08/08/nbc/. (n.d.).
https://medium.com/@kohlishivam5522/understanding-a-classification-report-for-your-machine-learning-model-88815e2ce397. (n.d.).
https://medium.com/towards-data-science/logic-and-implementation-of-a-spam-filter-machine-learning-algorithm-a508fb9547bd. (n.d.).
https://pythonmachinelearning.pro/text-classification-tutorial-with-naive-bayes/. (n.d.).
https://www.springboard.com/blog/data-analytics/naive-bayes-classification/. (n.d.).