DATA 7703 GROUP PROJECT REPORT
2020 Semester 2

SPAM CLASSIFICATION

Team 1: Runqi Lin Haochen Wang Jingbo Liu Haoyu Guo Xuhui He

We give consent for this to be used as a teaching resource.

1. Problem and significance

As we all know, the emergence of e-mail has greatly facilitated people's daily life. However, it also brings some troubles like privacy and spam. Nowadays, spam is getting worse, and we believe that most people have received spam e-mails, and some of them have been deceived by it. Besides, spam also could consume network resources and productivity, and cause the disclosure of personal information, so it is vital to find a way to solve this problem. Besides, some famous mailbox applications such as Gmail or Outlook have such function to divide e-mails into essential or less critical. These applications also can distinguish spam; however, some ham can be classified as spam, which usually leads to some unexpected problems.

Spam is a type of mail that is relatively easy to recognize and has some significant text characteristics as below:
Grammatical or spelling errors. Harassing e-mails are often loosely written and often have misspelt words and grammatical errors.
Leading words. Spam e-mails often have a lot of leading words, such as "winning", to induce users to click on relevant links or visit relevant websites.
The sender address is unknown. The e-mail address of the sender is unknown and does not belong to the internal suffix of the organization. Generally, the e-mail address of the sender with unfamiliar suffix often belongs to the category of spam.

The amount of spam is often huge and is often difficult to classify manually, so it needs to be classified automatically, and the machine learning method is the most commonly used in such case. In order to distinguish spam, the above characteristics can be used as criteria.

In the machine learning method, support vector machine and random forest algorithm can be used to train the model. In this project, the main target is to train an appropriate model for spam classification by adjusting parameters.

2. Solution

2.1 Processing Data

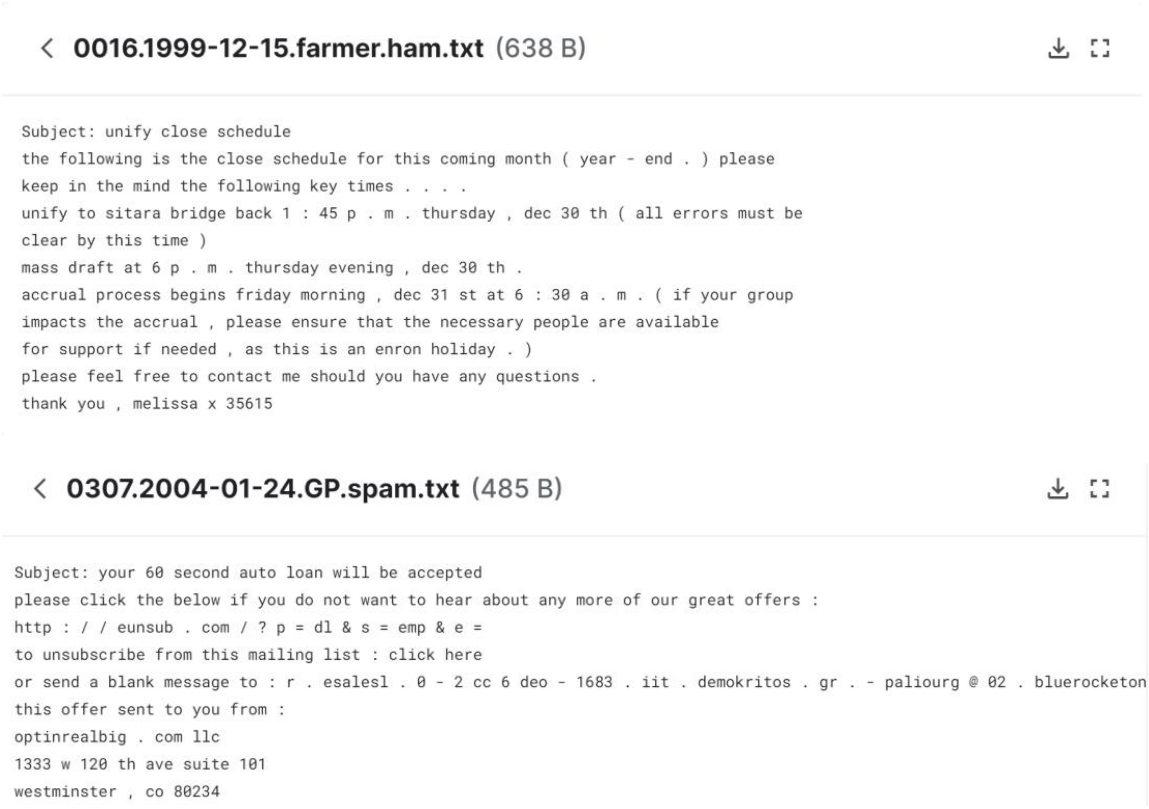
First of all, we should choose a decent dataset for our project. After discussion, we chose this dataset from kaggle. Here is the link below:

https://www.kaggle.com/wanderfj/enron-spam

This dataset has six folders, and each folder has a spam folder and ham(not spam) folder. There are have around 28000 files which are e-mail txt files.

Ham and spam are labelled for the e-mail text, ham represents regular e-mail, and spam represents spam.

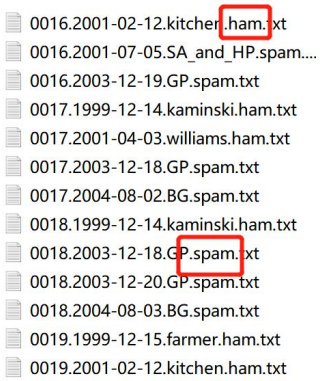
Here are the examples::



Secondly, because most machine learning algorithms require that the input data be numbers, and the content of these files contain text information. If we want to analyze

the dataset and model, we should convert these string or text data to numeric data.

Before we process data, we have to judge whether it is spam from the file name. So creating two variables in python which sign the spam or ham.




```
try:
    if 'spam' in file:
        label_sign = 1
        l_1 += 1
    elif 'ham' in file:
        label_sign = 0
        l_2 += 1
```

Thirdly, we should convert all words to lowercase letters with *lower()* in python. And we use the *Counter()*[1] function from *collections* package to count the words after conversion.

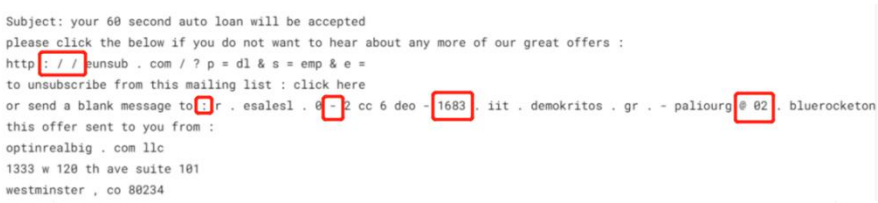
```
    lower_words = re.split(r"\W+", words.lower())
    count = collections.Counter(lower_words)

result = count.most_common(10)
```

Then we used the *most_common()*[2] to choose most ten words for each file. At first, we wanted to choose the most twenty words for each file, but because it would cause unnecessary calculations, we decided that the most ten words would be enough. And we should delete the invalid character or numbers from these data, like phone numbers or punctuation because these data are not helpful to our analysis.

---

[1] https://docs.python.org/2/library/collections.html#collections.Counter.most_common
[2] https://www.kite.com/python/docs/collections.Counter.most_common

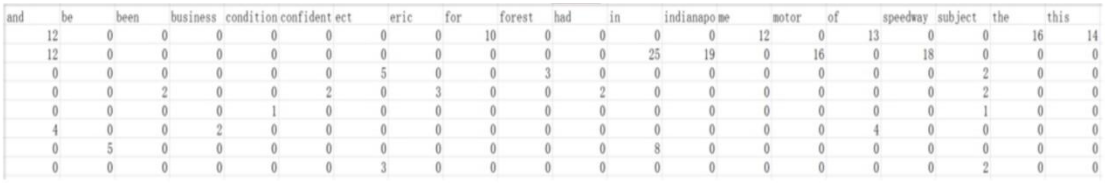



```
('1431', 3), ('1999', 3)
```

Next, the conversion method used here becomes the Bag-of-Words model, whose core purpose is to divide a string of text into a word frequency distribution matrix. The tool to be used this time, which is the *CountVectorizer()*[3] class of *sklearn* which can convert a collection of text documents to a matrix of token counts. This implementation produces a sparse representation of the counts using scipy.sparse .csr_matrix. The two methods *fit_transform()* and *get_feature_names()* are mainly used. The *fit_transform()* function learn the vocabulary dictionary and return document-term matrix. And the *get_feature_names()* function array mapping from feature integer indices to feature name. And we also used *todense()* and *tolist()* functions to change the sparse matrix to list and added the label variables in the list at last in order to write a CSV file .

Finally, we wrote the above content into a new CSV file. There are 7529 columns which indicate the number of occurrences of the word in each email. The last column has the labels for prediction: 1 for spam, 0 for not spam.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html



and	be	been	business	condition	confident	ect	eric	for	forest	had	in	indianapo	me	motor	of	speedway	subject	the	this
12	0	0	0	0	0	0	0	10	0	0	0	0	12	0	13	0	0	16	14
12	0	0	0	0	0	0	0	0	0	0	25	19	0	16	0	18	0	0	0
0	0	0	0	0	0	5	0	0	3	0	0	0	0	0	0	0	2	0	0
0	0	2	0	0	2	0	3	0	0	2	0	0	0	0	0	0	2	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
4	0	0	2	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0
0	5	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0
0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	2	0	0

confident	ect	eric	for	forest	had	in	indianapo	me	motor	of	speedway	subject	the	this	to	you	your	label
0	0	0	10	0	0	0	0	12	0	13	0	0	16	14	25	16	10	1
0	0	0	0	0	0	25	19	0	16	0	18	0	0	0	16	0	0	1
0	5	0	0	3	0	0	0	0	0	0	0	2	0	0	0	0	0	0
2	0	3	0	0	2	0	0	0	0	0	0	2	0	0	4	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	1	3	1
0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	3	1
0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	7	0	0	0
0	3	0	0	0	0	0	0	0	0	0	0	2	0	0	4	0	0	0

The remaining 7528 columns are the most common words in all the emails, after excluding the non-alphabetical characters/words(like the phone number, punctuation). For each row, the count of each word(column) in that email(row) is stored in the respective cells. Thus, information regarding all 9000 emails (We initially chose all 28,000 emails, but our computer processing efficiency was reduced due to a large amount of data. After discussion, it was finally reduced to 9,000 emails.) is stored in a compact data frame rather than as separate text files.

We used SVM and random forest algorithms for training, respectively. When using the two algorithms to model separately, multiple sets of parameter combination schemes are used. The model performance results are analyzed, and finally, the best performing model parameters will be returned. The model with the best performance of the two algorithms is extracted and analyzed and compared (including the identification of essential variables, etc.).

2.2 Support Vector Machines

In terms of SVM, it is an approach that maps the vector to a higher-dimensional space, and a maximum separation hyperplane is established in this space. Two hyperplanes parallel to each other are built on both sides of the hyperplane that separates the data. The separating hyperplane maximizes the distance between two parallel hyperplanes. The most significant reason why we choose SVM to deal with the spam e-mail is that

emails are character, non-linear data, which are complicated and through SVM, it can be converted to linear data, which is easier to deal with. The linear separation only needs to be a straight line or a plane, while non-linear includes many different cases that do not have a standard to deal. The other reason is that SVM can get much better results than other algorithms on the small sample training set. The goal of SVM is to minimize structural risk, not to minimize empirical risk. Therefore, through the concept of margin, a structured description of data distribution is obtained, which reduces the requirements for data scale and data distribution.

2.3 Random Forests

As for random forest, it is a method of using multiple decision trees to distinguish and classify data. While classifying the data, the random forest can also specify the estimated value of various variables to evaluate the classification. For classifying e-mails, it can not only generate highly accurate classifiers, but it also can handle data with massive features and no need to make the feature selection. Another reason is that the random forest has a fast training speed, and during the training process, it can be able to detect the interaction between the features. Although random forests have been proved to be over-fitting in some noisy classification or regression problems, we believe that in this project, the noise will not affect much.

3. Findings

3.1 Support Vector Machines

In terms of SVM, I use the package of SVM and K-Fold to build the SVM model, I also divide the data set into ten fold randomly, nine for training data, one for testing data.

For each model, use each fold as testing data once and calculate the average value to

improve accuracy. In addition, use accuracy_score, precision_score, recall_score, f1_score to evaluate the performance of the model.

In this function, I change three different parameters: kernel, C and gamma to adjust the model.

Kernel means the kernel function that is used to convert the non-linear problems to a linear problem. For this parameter, I choose three kernel function: linear, rbf(Gaussian kernel function) and poly. Among these three functions with other default parameters, linear perform best and poly perform worse. The linear function is mainly used for linearly separable situations. The time cost of linear is the lowest among the 3. RBF function is used in the case of linear inseparability. There are many parameters, and the classification result is very dependent on the parameters. However, the process of adjusting the parameters is time costing. For poly, this requires many labels in the training set; in this case, there is only one label to classify. Therefore, poly performs worst among the three kernel function.

kernel	accuracy	precision	recall	F1
linear	0.9366666666666	0.9222462203023	0.953125	0.9374313940724
rbf	0.7577777777778	0.6866883116883	0.9441964285714	0.7951127819548
poly	0.53	0.5143513203214	1.0	0.6793025018953

As for gamma, which means the coefficient of rbf and poly, the default is auto, and it will use the one north of features, for this spam-ham classification, the default gamma is 0.5. In this case, I choose auto, 0.1 and 10 for gamma. The larger the gamma, the fewer support vectors, and the smaller the gamma, the more support vectors, which will increase the acc. Over 10, the change of gamma will not affect the model a lot.

Gamma	accuracy	precision	recall	F1
auto	0.7577777777778	0.6866883116883	0.9441964285714	0.7951127819548
0.1	0.7888888888889	0.9328859060402	0.6205357142857	0.745308310991

10	0.5833333333334	0.5443499392466	1.0	0.7049567269866

In the aspect of C, the penalty coefficient is used to control the penalty coefficient of the loss function. When C increases from default (1) to 10 and 100, the acc is increasing too. It means it will fit this spam classifier better.

C	accuracy	precision	recall	F1
1	0.7577777777778	0.6866883116883	0.9441964285714	0.7951127819548
10	0.8666666666667	0.8141762452107	0.9486607142857	0.8762886597938
100	0.9244444444444	0.8909465020576	0.9665178571428	0.9271948608137

To make a brief conclusion, the linear kernel function fits the case best because the case is linear separable and there are many features. In addition, the increase of C will also increase the acc of the model while it may cause overfitting.

3.2 Random Forests

As we know, the random forest is good at deal with high dimensional data. Firstly we used the random forest to classify spam and ham. We divided the entire dataset into train set (including attributes vectors and targeted values) and test set with the ratio of 7 to 3 by *train_test_split* function, including attributes vectors (the number of each word contained in emails) and targeted values (the email is ham or spam). To maintain the results of the subset, we keep the random state at 65.

In this section, we use Accuracy, Precision, Recall and F1 score to measure the performance of the model. Accuracy is the proportion of the number of samples that are correctly predicted to the total number of samples. Precision is the ratio of samples that are true positive to samples that are predicted to be positive. Recall indicates how many positive examples in the sample are predicted correctly. F-Measure is the weighted harmonic average of Precision and Recall.

The model is trained by *RandomForestClassifier* class and fitted by the train set. The attributes vectors of the test set are inserted in the constructed model to predict whether the emails represented by the vectors are spam. With default parameters (except criterion='entropy', random_state=42), the performance is good, all those measurement values are larger than 0.9.

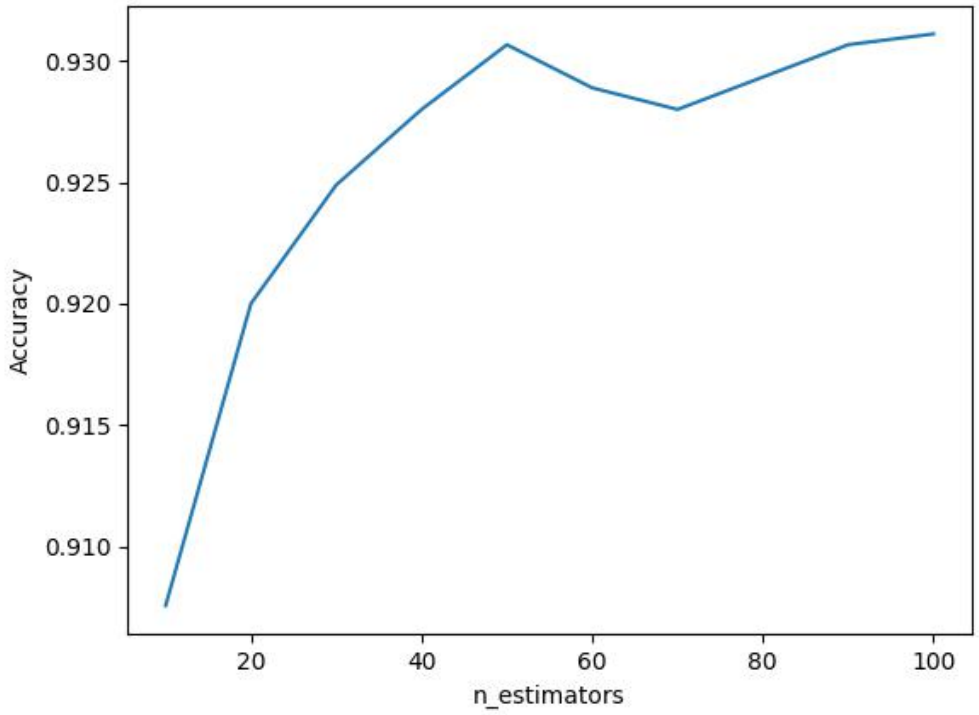
Accuracy score: 0.9182222222222223

Precision score: 0.912751677852349

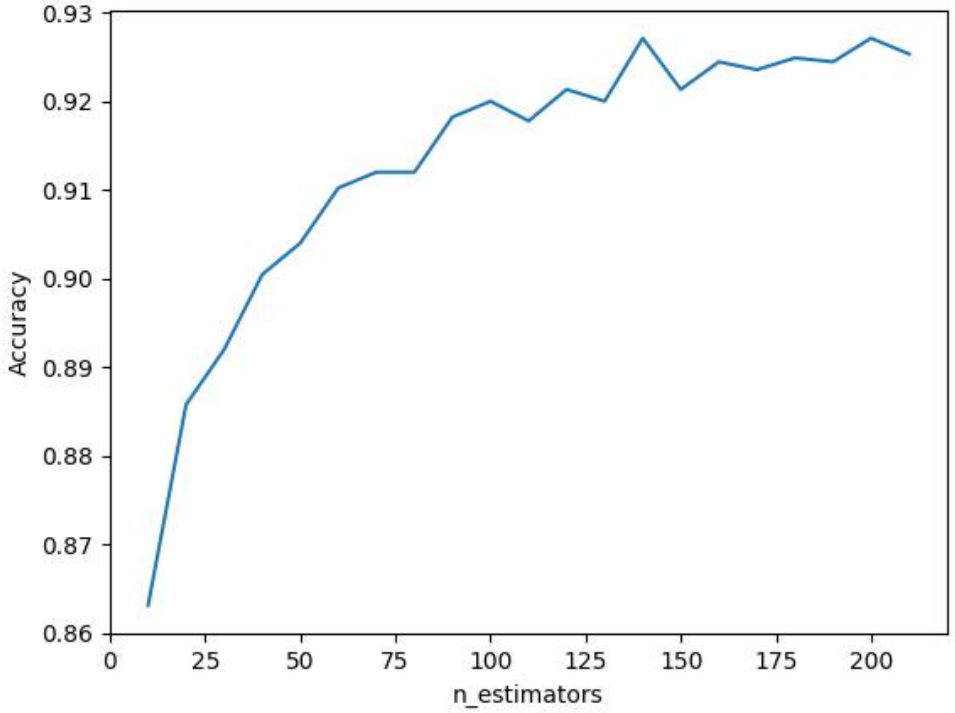
Recall score: 0.9315068493150684

F1 score: 0.9220338983050848

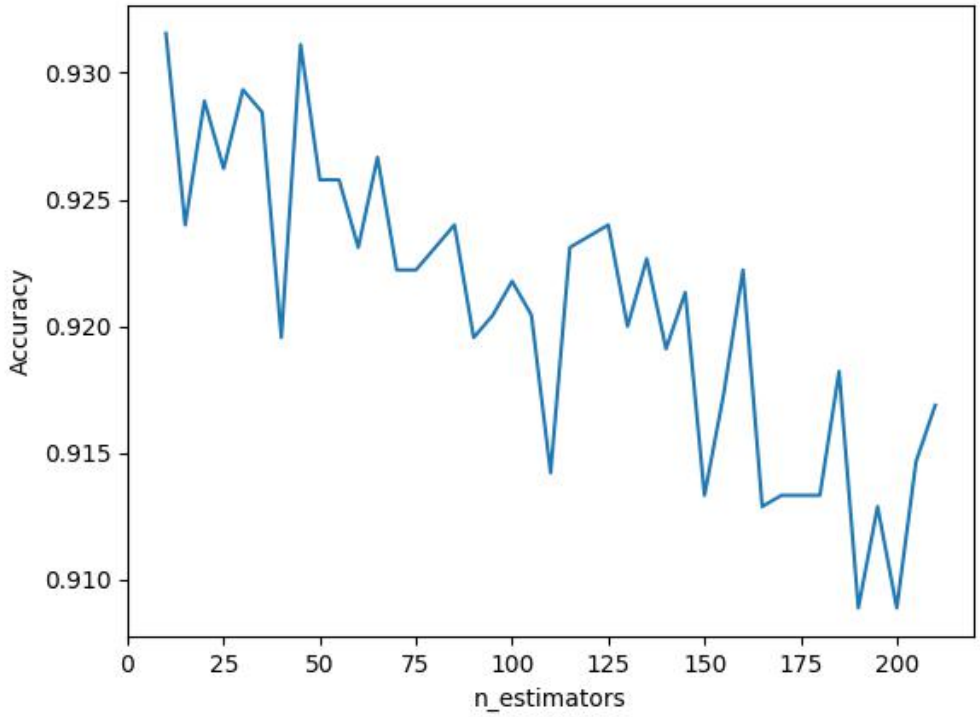
To get better performance, we adjust several parameters sequentially: the number of estimators, max depth and max features. To find out the impact of the number of estimators on performance, we kept the other two parameters default and tried the different number of estimators, start from 10 to 100 (Step size is 10). The accuracy increases significantly from 10 to 50. When the number of estimators equal to 50, the performance is the best, the accuracy reaches the highest point at about 0.93.



Because there are a large number of data in our dataset, it is essential to set appropriate max depth and max feature. From the result of the above step, the number of estimators remains at 50. The range of max depth value is from 10 to 210 (step is 10). There is a significant increase in accuracy from 10 to 140. When the max depth is larger than 140, the accuracy increases slowly, marginal growth rate decreases. Therefore we believe the max depth equal to 140 is appropriate.



Finally, we kept n_estimators and max_depth at 50 and 140, which are considered as the best parameters combination in this dataset. The range of max depth value is from 10 to 210 (step is 5). When the number of max features is 10, the accuracy is the highest. In fact, there is no obvious functional relationship between accuracy and max_features; there is a fluctuation in the figure. Therefore, we take the best of the test values. If the step size becomes smaller, there should be a better parameter.



Now we get the best parameters combination (n_estimitors=50, max_depth=140, max_features=10) for our model. According to the test results of the new model, we can see its performance is better than default parameters.

Accuracy score: 0.9315555555555556

Precision score: 0.9031993437243643

Recall score: 0.9683377308707124

F1 score: 0.9346349745331068

In order to prevent overfitting for the above training set, we will use 10-fold cross-validation to measure the accuracy of the random forest model again. This can also reduce the deviation of the measurement. We used the KFold function in sklearn to divide the data set into ten parts randomly (random_state=42) and take turns using nine parts as training data and 1 part as test data for testing. By using the same random forest function, ten models are constructed. Each test will get the corresponding accuracy. We keep the parameters maintained at (n_estimitors=50,

max_depth=140, max_features=10). The average accuracy often-times modelling is the final consequence which is 0.9288811024595229.

In summary, the model of random forest can accurately classify all emails; it can efficiently filter spam and will not filter excessively to cause ham emails to be mistakenly regarded as spam.

4. Limitations

There is not a specific method to choose the kernel function. Since SVM uses quadratic programming to solve support vectors, and solving quadratic programming involves the calculation of m-order matrices (m is the number of samples), when the number of m is large, the storage and calculation of the matrix will consume a lot of storage space and computing time..

Another limitation is that our group uses the number of occurrences of the word to classify ham and spam. It means that the more spam keywords contained in an e-mail, the more likely it is to be spam, so a long e-mail might contain more keywords, and is, therefore, more likely classified as spam. It makes the predicted results of the models be related to the e-mail length. We believe using the frequency of the word could improve our model's performance; the models will be no longer related to the length. For random forest, usually, each feature should be independent of each other. However, in our random forest model, the individual words are not necessarily completely independent of each other. Due to some fixed collocations, some meaningful keywords in the e-mail may always be followed by certain prepositions; these prepositions might affect the interpretability of the model. Therefore, reducing the weight of prepositions and increasing the weight of words in ham will also help improve the accuracy of our model. The limitation of the support vector machine is that it is challenging to choose the kernel function, and the cost is relatively expensive.

Moreover, there are many other ways which can classify spam-like logistic regression, knn, Naive Bayes and Multilayer Perceptron. Due to time constraints, we do not test them all, if we have enough time we will find the best way to classify spam. The other limitation is we only use accuracy, precision, recall and F1 score to estimate the performance. However, the filtering time and training time should also be considered.

In addition, in the modelling process, initially, our data volume was too large, which caused the program to run extremely slowly. After we randomly deleted half of the data, the program was running more smoothly. Also, adjusting the parameters of each model will help to obtain the best performance of the model, establish a graphical relationship between the changes of each parameter and the accuracy of the model, and understand the approximate relationship between the two to obtain the optimal parameters.

5. Conclusion

In conclusion, spam is a severe problem and using an appropriate model to classify spam is an excellent way to deal with it. After we test using random forest and support vector machine to train spam classification model can achieve good performance. However, both of them has their problems, like the cost of SVM is relatively high, and it is difficult to ensure the subtree of random forest is independent. So in the future, we can focus on those aspects to improve our model.