DETECTION OF SPAM IN EMAILS

IST 664 – Natural Language Processing

Professor Lin

Abstract

This analysis investigates the efficiency of machine learning techniques in determining spam from authentic emails using a dataset derived from the Enron corpus, enhanced with additional spam for balanced classification.

Arti Ravi Garg- Data pre-processing and Feature Engineering,
Ryan Richardson- Modeling,
Nolan Arendt- Presentation and Iteration
aravigar@syr.edu, ryrichar@syr.edu, nnarendt@syr.edu

INTRODUCTION

Emails are a large part of our daily lives, used by everyone from individuals to big companies to communicate. However, not all emails we receive are useful; some emails we receive are spam, which means they are unwanted and potentially harmful. Our project focuses on finding a way to determine whether emails are authentic in nature, or if they are spam and are of no value. We used a special collection of emails for this, which includes thousands of real emails, along with spam emails added for research purposes. This collection of both spam and authentic emails helps us understand how spam looks different from authentic emails so that we can teach a computer program to determine the difference with a high accuracy rate.

The initial phase of our analysis was to clean and organize the emails, making it easier for the computer to understand them, and for the algorithms we build to perform properly. We looked for common patterns and features that could help distinguish authentic emails from spam emails, like specific words or phrases that are more likely to appear in spam. Then, we used a method called the Naïve Bayes classifier, a type of machine learning classifier that learns from examples, to see if it could learn to identify spam based on the patterns we found. We checked our model's performance by using multiple evaluation techniques that help us understand how it performs and what it needs further improvement on, making sure it can accurately determine authentic emails from spam emails.

Following the initial analysis, our exploration extended beyond the Naïve Bayes classifier to include a variety of machine learning models. This expanded approach allowed us to compare the effectiveness of different classifiers in classifying between spam and authentic emails. By not limiting our analysis to a single algorithm, we deepened our research with a wider perspective on how different algorithms handle the complexities of email classification. Each model underwent thorough testing and evaluation to identify the model that offers the highest accuracy in spam detection. This process of trial, analysis, and refinement highlights the attention to detail in our approach, highlighting our commitment to improving email security. Through this thorough examination of multiple models, our project moves a step closer to enhancing the efficiency and safety of email communication for users everywhere.

Pre-Processing of Data

1. **Data Preparation:** Initially, we had approx. 3700 ham mail (.txt) files and 1500 spam mail (.txt) files. To ensure unbiased analysis and prediction, we balanced the data by selecting an equal number of files from both classes, resulting in 1500 mails from each category.
2. **Tokenization:** Following data balancing, we tokenized the emails. This process involved creating a list of email documents, where each document comprised a

list of words (tokens) extracted from the email text, along with its corresponding label ("spam" or "ham").

3. **Random Shuffling:** To introduce randomness and prevent any inherent bias in the data, we shuffled the list of email documents. This step ensures that the order of emails does not influence subsequent analyses or predictions. Finally, we inspected an example email to verify the format of our processed list.
4. **Lowercasing Words:** Additionally, we converted all tokens to lowercase to ensure consistency and improve the quality of our analysis and predictions. This preprocessing step helps in standardizing the text data and reduces the complexity of subsequent operations.

Outcome: After these preparations, we obtained the total number of emails in our dataset along with a glimpse of the first element of the list, showcasing our structured data format.

```
Total email for experiment :3000
Example email (['Subject', ':', 'ei', '315', '/', '329', 'revised', 'availability', 'effective', '6', '/', '17',
'/', '00', '(', 'tennessee', 'p', '/', 'l', ')', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-',
'-', '-', '-', '-', '-', '-', '-', '-', '-', 'forwarded', 'by', 'ami', 'chokshi', '/', 'corp', '/', 'enron', 'on', '0
6', '/', '16', '/', '2000', '12', ':', '24', 'pm', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-',
'-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', "'", 'steve', 'holmes', '``', 'on', '06',
'/', '16', '/', '2000', '10', ':', '41', ':', '55', 'am', 'to', ':', ',', 'cc', ':', 'subject', ':', 'ei', '315',
'/', '329', 'revised', 'availability', 'effective', '6', '/', '17', '/', '00', '(', 'tennessee', 'p', '/', 'l', ')',
'i', 'just', 'received', 'notification', 'from', 'the', 'eugene', 'island', '315', 'platform', 'that', 'they', 'wil
l', 'be', 'shut', '-', 'in', 'for', '2', '-', '3', 'days', 'while', 'the', 'drilling', 'rig', 'moves', 'off', 'locat
ion', '.', 'i', 'will', 'notify', 'you', 'monday', 'morning', 'of', 'its', 'status', '.', 'thanks', ',', 'steve', '-
', 'ei', '315329', 'reveffo', '61700', '.', 'xls'], 'ham')
```

Model Creation and Results:

1. **Unigram Approach(Naïve Bayes NLTK):** In this strategy, we compiled a list of frequently occurring words found in emails and constructed a feature set comprising these words. This feature set was then utilized in our model to facilitate classification tasks.

Top 50 frequent words.

```
['-', '.', '/', ',', ':', 'the', 'to', 'ect', 'and', 'of', '@', 'a', 'for', '?', 'you', 'in', 'this', 'is', 'hou',
'subject', 'on', 'i', "'", ')', '=', '(', 'enron', '!', 'be', '2000', 'your', 'that', 'with', '_', 'from', 'will',
'have', 's', 'we', 'as', 'are', 'it', '$', '>', 'or', '3', 'at', 'not', 'by', '``']
```

For this model, we selected the top 2000 most frequent words from our dataset. We divided our data into a training set comprising 90% of the total emails and a test set comprising the remaining 10%. The model was trained on the training set and then evaluated for accuracy using the test set.

Result:

Accuracy:96.33%

Most Informative Features:

```
Most Informative Features
             V_forwarded = True              ham : spam   =    297.4 : 1.0
                   V_hou = True              ham : spam   =    262.8 : 1.0
                   V_ect = True              ham : spam   =    165.1 : 1.0
                   V_nom = True              ham : spam   =    117.9 : 1.0
            V_nomination = True              ham : spam   =     84.6 : 1.0
                 V_steve = True              ham : spam   =     68.2 : 1.0
                V_farmer = True              ham : spam   =     63.4 : 1.0
                 V_susan = True              ham : spam   =     57.8 : 1.0
                    V_cc = True              ham : spam   =     56.4 : 1.0
                 V_lloyd = True              ham : spam   =     51.9 : 1.0
                V_jackie = True              ham : spam   =     47.3 : 1.0
                   V_ami = True              ham : spam   =     47.2 : 1.0
                  V_2005 = True             spam : ham    =     46.6 : 1.0
                V_howard = True              ham : spam   =     43.4 : 1.0
               V_follows = True              ham : spam   =     42.1 : 1.0
                V_stacey = True              ham : spam   =     40.2 : 1.0
                 V_hanks = True              ham : spam   =     39.5 : 1.0
                 V_super = True             spam : ham    =     38.5 : 1.0
                 V_unify = True              ham : spam   =     36.2 : 1.0
                 V_vance = True              ham : spam   =     35.8 : 1.0
                 V_brand = True             spam : ham    =     35.7 : 1.0
```

These features indicate the likelihood of an email being spam or ham based on the presence or absence of certain words.

Cross-Validation Results- 5 folds

Mean Accuracy:96.7%

Confusion Matrix:

Precision,Recall,F1:

```
      |             s |
      |      h      p |
      |      a      a |
      |      m      m |
------+---------------+
 ham  | <46.7%>  3.3% |
spam  |   0.0% <50.0%>|
------+---------------+
(row = reference; col = test)
```

```
          Precision       Recall           F1
  spam          0.999       0.939       0.968
  ham           0.935       0.999       0.966
```

The mean accuracy of our classification model is 96.7%. This indicates that our model correctly predicts the class labels for approximately 96.7% of the test instances. Looking

at the confusion matrix, we observe that the majority of ham emails are correctly classified, with a precision of 93.5% and recall of 99.9%. Similarly, for spam emails, the precision is 99.9%, and recall is 93.9%. These metrics suggest that our model performs well in distinguishing between ham and spam emails, with high precision and recall values for both classes. Overall, the F1 score, which balances precision and recall, is high for both ham and spam classes, indicating robust performance across different evaluation metrics.

2. **Stop Words Removal Approach-** In this approach, we filter out common stop words from the text data, which are typically frequent words that often do not carry significant meaning for the analysis.

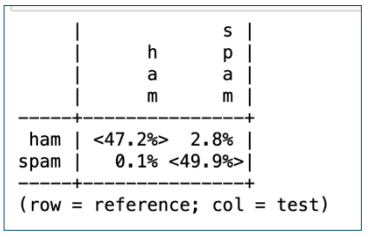
For this model also, we selected the top 2000 most frequent words after removing the stop words of 'English'. from our dataset. We divided our data into a training set comprising 90% of the total emails and a test set comprising the remaining 10%. The model was trained on the training set and then evaluated for accuracy using the test set.

Result: **Accuracy:97.00%**

Cross-Validation Results- 5 folds

Mean Accuracy:97.03%

Confusion Matrix:



Precision,Recall,F1:

	Precision	Recall	F1
spam	0.997	0.946	0.971
ham	0.943	0.997	0.970

The mean accuracy of our classification model is 97.03%. This indicates that our model correctly predicts the class labels for approximately 97.03% of the test instances. Looking at the confusion matrix, we observe that the majority of ham emails are correctly classified, with a precision of 94.3% and recall of 99.7%. Similarly, for spam emails, the precision is 99.7% and recall is 94.6%. These metrics suggest that our model performs well in distinguishing between ham and spam emails, with high precision and recall values for both classes. Overall, the F1 score, which balances precision and recall, is high for both ham and spam classes, indicating robust performance across different evaluation metrics.

3. **The POS tagging approach**: involves labeling words in a text with their corresponding parts of speech, such as nouns, verbs, adjectives, etc.

Our POS Tagged Data


```
([[('Subject', 'JJ'), (':', ':'), ('ei', 'NN'), ('315', 'CD'), ('/', '$'), ('329', 'CD'), ('revised', 'VBN'), ('avai
lability', 'NN'), ('effective', 'JJ'), ('6', 'CD'), ('/', 'RB'), ('17', 'CD'), ('/', 'JJ'), ('00', 'CD'), ('(',
'('), ('tennessee', 'NN'), ('p', 'NN'), ('/', 'NNP'), ('l', 'NN'), (')', ')'), ('-', ':'), ('-', ':'), ('-', ':'),
('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-',
':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'),
('forwarded', 'VBN'), ('by', 'IN'), ('ami', 'NN'), ('chokshi', 'NN'), ('/', 'NNP'), ('corp', 'NN'), ('/', 'NNP'),
('enron', 'NN'), ('on', 'IN'), ('06', 'CD'), ('/', 'JJ'), ('16', 'CD'), ('/', 'JJ'), ('2000', 'CD'), ('12', 'CD'),
(':', ':'), ('24', 'CD'), ('pm', 'NN'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-
', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'),
('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-', ':'), ('-',
':'), ('-', ':'), ('```', '```'), ('steve', 'VB'), ('holmes', 'VBZ'), ('```', '```'), ('on', 'IN'), ('06', 'CD'), ('/',
'JJ'), ('16', 'CD'), ('/', 'JJ'), ('2000', 'CD'), ('10', 'CD'), (':', ':'), ('41', 'CD'), (':', ':'), ('55', 'CD'),
('am', 'VBP'), ('to', 'TO'), (':', ':'), (',', ','), ('cc', 'NN'), (':', ':'), ('subject', 'NN'), (':', ':'), ('ei',
'NN'), ('315', 'CD'), ('/', '$'), ('329', 'CD'), ('revised', 'VBN'), ('availability', 'NN'), ('effective', 'JJ'),
('6', 'CD'), ('/', 'RB'), ('17', 'CD'), ('/', 'JJ'), ('00', 'CD'), ('(', '('), ('tennessee', 'NN'), ('p', 'NN'),
('/', 'NNP'), ('l', 'NN'), (')', ')'), ('i', 'NN'), ('just', 'RB'), ('received', 'VBN'), ('notification', 'NN'), ('f
rom', 'IN'), ('the', 'DT'), ('eugene', 'NN'), ('island', 'NN'), ('315', 'CD'), ('platform', 'NN'), ('that', 'IN'),
('they', 'PRP'), ('will', 'MD'), ('be', 'VB'), ('shut', 'VBN'), ('-', ':'), ('in', 'IN'), ('for', 'IN'), ('2', 'C
D'), ('-', ':'), ('3', 'CD'), ('days', 'NNS'), ('while', 'IN'), ('the', 'DT'), ('drilling', 'NN'), ('rig', 'NN'),
('moves', 'VBZ'), ('off', 'RP'), ('location', 'NN'), ('.', '.')], [('i', 'NN'), ('will', 'MD'), ('notify', 'VB'),
('you', 'PRP'), ('monday', 'JJ'), ('morning', 'NN'), ('of', 'IN'), ('its', 'PRP$'), ('status', 'NN'), ('.', '.')],
[('thanks', 'NNS'), (',', ','), ('steve', 'VBP'), ('-', ':'), ('ei', 'NN'), ('315329', 'CD'), ('reveffo', 'NN'), ('6
1700', 'CD'), ('.', '.')], [('xls', 'NN')]], 'ham')
```


After this, we created our feature set by counting the tag of each e-mail.


```
({'JJ': 10, ':': 59, 'NN': 31, 'CD': 26, '$': 2, 'VBN': 5, 'RB': 3, '(': 2, 'NNP': 4, ')': 2, 'IN': 9, '``': 2, 'V
B': 3, 'VBZ': 2, 'VBP': 2, 'TO': 1, ',': 2, 'DT': 2, 'PRP': 2, 'MD': 2, 'NNS': 2, 'RP': 1, '.': 3, 'PRP$': 1, 'subje
ct': 2, 'ei': 3, 'revised': 2, 'availability': 2, 'effective': 2, 'tennessee': 2, 'p': 2, 'l': 2, 'forwarded': 1, 'b
y': 1, 'ami': 1, 'chokshi': 1, 'corp': 1, 'enron': 1, 'on': 2, 'pm': 1, 'steve': 2, 'holmes': 1, 'am': 1, 'to': 1,
'cc': 1, 'i': 2, 'just': 1, 'received': 1, 'notification': 1, 'from': 1, 'the': 2, 'eugene': 1, 'island': 1, 'platfo
rm': 1, 'that': 1, 'they': 1, 'will': 2, 'be': 1, 'shut': 1, 'in': 1, 'for': 1, 'days': 1, 'while': 1, 'drilling':
1, 'rig': 1, 'moves': 1, 'off': 1, 'location': 1, 'notify': 1, 'you': 1, 'monday': 1, 'morning': 1, 'of': 1, 'its':
1, 'status': 1, 'thanks': 1, 'reveffo': 1, 'xls': 1}, 'ham')
```



```
     |              s |
     |      h       p |
     |      a       a |
     |      m       m |
-----+----------------+
 ham | <47.1%>  2.9%  |
spam |   0.0% <50.0%>|
-----+----------------+
(row = reference; col = test)
```


Then we applied Naïve Bayes approach for Model creation.

Result: **Accuracy:97.00%**

Cross-Validation Results- 5 folds

Mean Accuracy:97.06%

Confusion Matrix:

Precision,Recall,F1:

	Precision	Recall	F1
spam	0.999	0.945	0.971
ham	0.942	0.999	0.970

The classifier achieved an accuracy of 97%. In the confusion matrix, 47.1% of ham emails were correctly classified, while 2.9% were incorrectly classified as spam. For spam emails, 50% were correctly classified, and none were incorrectly classified as ham. The precision for spam emails is 99.9%, indicating that the vast majority of emails classified as spam were indeed spam. The recall for ham emails is also high at 99.9%, indicating that almost all ham emails were correctly identified. The F1 score, which balances precision and recall, is 97.1% for spam and 97% for ham, indicating overall good performance of the classifier.

Additionally, we utilized an advanced technique to forecast spam emails.

4. Logistic Regression Classifier: Results

```
Best: 0.974167 using {'max_iter': 100, 'tol': 0.01}
Test set accuracy:  0.9766666666666667
              precision    recall  f1-score   support

         ham       0.99      0.96      0.97       279
        spam       0.96      0.99      0.98       321

    accuracy                           0.98       600
   macro avg       0.98      0.98      0.98       600
weighted avg       0.98      0.98      0.98       600

Best: 0.974167 using {'max_iter': 100, 'tol': 0.01}
```

The logistic regression model achieved an accuracy of approximately 97.67% on the test set, indicating its effectiveness in distinguishing between ham and spam emails. With a precision of 99% for ham and 96% for spam, the model demonstrates a high level of correctness in classifying both types of emails. Additionally, the recall values are also high, indicating that the model effectively captures most of the relevant instances of both ham and spam emails. Overall, the logistic regression model shows robust performance in classifying emails, with balanced precision and recall scores for both classes, contributing to its high accuracy.

5. Random Forest Classifier: Results

```
Best: 0.964167 using {'criterion': 'gini', 'max_features': 'sqrt', 'min_samples_split': 0.1, 'n_estimators': 200}
Test set accuracy:  0.96
              precision    recall  f1-score   support

         ham       0.99      0.92      0.96       279
        spam       0.94      0.99      0.96       321

    accuracy                           0.96       600
   macro avg       0.96      0.96      0.96       600
weighted avg       0.96      0.96      0.96       600
```

The Random Forest model achieved an accuracy of 96% on the test set, indicated a lower effectiveness in distinguishing ham and spam emails compared to other model options. While the model performs well, it is not the preferred option available.

6. Gradient Boosting Classifier: Results

```
Fitting 5 folds for each of 81 candidates, totalling 405 fits
Best parameters found:  {'learning_rate': 0.2, 'max_depth': 3, 'n_estimators': 300, 'subsample': 1.0}
Best cross-validated score:  0.9745833333333334
Test set accuracy:  0.9766666666666667
              precision    recall  f1-score   support

         ham       1.00      0.95      0.97       279
        spam       0.96      1.00      0.98       321

    accuracy                           0.98       600
   macro avg       0.98      0.98      0.98       600
weighted avg       0.98      0.98      0.98       600
```

The Gradient Boosting classifier achieved an accuracy of 97.67% on the test set, tying with the Logistic Regression classifier. However, the Gradient Boosting classifier additionally achieved perfect precision on classifying ham emails. Providing an intriguing model option.

7. Support Vector Machine Classifier: Results

```
Best parameters found:  {'C': 100, 'gamma': 'auto', 'kernel': 'rbf'}
Best cross-validated score:  0.9737500000000001
Test set accuracy:  0.9683333333333334
              precision    recall  f1-score   support

         ham       0.99      0.94      0.97       279
        spam       0.95      0.99      0.97       321

    accuracy                           0.97       600
   macro avg       0.97      0.97      0.97       600
weighted avg       0.97      0.97      0.97       600
```

The Support Vector Machine classifier achieved an accuracy of 97.375% on the test, scoring slightly lower than the Logistic Regression and Gradient Boosting classifier models. As of this run, this model would not be normally investigated, however, there are some additional considerations noted in the evaluation section that should give one pause on this model.

Evaluation

As we examine various model performances, considerations should be made for compute resources compared to model performance. Additionally, tolerance levels for model performance on different email classifications should be weighed. We used a Naïve Bayes classifier from NLTK as a baseline model to inform which preprocessing technique yielded the best results and then used that preprocessing methodology for further investigation into various model comparisons from scikit-learn. In total, we a standard suite of classification models – Naïve Bayes from NLTK, and then Logistic Regression, Random Forest, Gradient Boosting, and Support Vector Machines from sci-kit learn.

Our cross-validated Naïve Bayes models yielded 96.70% mean accuracy when all unigrams were considered, 97.03% mean accuracy when stopwords were removed, and 97.06% mean accuracy when POS tag counts were added to the feature set for each email. We noted that in some tests, data with stopwords removed performed better than the POS tag count data, and further evaluation should include more rigorous testing with a higher number of folds in cross validation. Further, there are additional tradeoffs beyond accuracy between the two datasets. Notably, stopword removal reduces the number of features in the datasets as opposed to increasing features by included POS tag counts. This means that the reduced data complexity from removing stopwords also reduces the computational load when training additional models. Given the negligible difference between the two models, we opted to focus further efforts on the data without stopwords.

For the next set of models, Logistic Regression provides a good starting point model – it's fast, easy to tune, and in this case, performs exceptionally well – at 97.67% accuracy. The accuracy-to-train-time cost is most favorable for this model and for most use cases, this would probably be the model of choice. Other model options might have slightly better performances in the right circumstances and under the right hyperparameters – but the computational tradeoffs are significant.

The other particularly noteworthy models are Gradient Boosting. While the mean accuracy of this model ties with Logistic Regression, it has better precision on ham emails – in fact, in this run it has perfect precision on ham emails. At an enterprise level, where at scale every minor improvement can have large implications, this model would better assure that a ham email would never be incorrectly filtered and missed. However, it should be noted that this improvement has significant computational cost and one should consider whether the tradeoff is worthwhile (for comparison, on my relatively low-powered machine, a GridSearchCV on a logistic regression model took about 1 minute, whereas the GridSearchCV on GradientBoost took more than an hour and a half).

Lastly, a Google search suggests that, with appropriate preprocessing – a Support Vector Machine actually performs the best on this dataset, and, while it did not yield the best results for us, in some of our runs, we did experience similar 100% precision on ham email identification, and it might serve as a particularly valuable approach if one wanted to minimize the likelihood of misclassifying a valid email. Further testing should also investigate this model. Note that we did also run a Random Forest classifier as well, however, it has no performance benefits over a Logistic Regression classifier and has higher computational cost.

Challenges

We should note a couple of challenges we faced during the project that merit consideration for future iterations. First, we note that the model tends to overfit the training data and more robust model testing and evaluation methods are needed. For the sake of rapid iteration, we only used 5-fold cross validation to help prevent this, although future iterations may want to employ 10 or more folds for more robust model evaluation.

Additionally, we noted above that the computational needs of some of the models is pretty strenuous. While we have already implemented some feature selection to reduce dimensionality, future investigations should attempt to further reduce this dimensionality by removing additional unnecessary features. If enough unnecessary features could be removed, it might enable smaller clients to take advantage of some of the more powerful methodologies, as well as enable faster product iteration.

Conclusion

In summary, we successfully built several models capable of effectively classifying spam emails. We discovered that a simple Logistic Regression classification model would probably work well enough for smaller clients with lower computational resources and lower email volume (and consequently, lower risk at scale with misclassifications). Larger, enterprise level clients would likely want to deploy a more powerful solution – like a Gradient Boosting model or a Support Vector Machine – to ensure that valuable emails across the company were not misclassified and missed. At scale, these kinds of misclassifications could lead to significant drops in customer service and, potentially, revenue.

We do note that these models represent a minimum viable product. Further testing is needed for future product iterations. Many of the spam emails in the dataset are obvious, and spam threats have evolved over time with escalating risks through actors deploying ransomware attacks. We suggest that real example spam phishing/spear phishing emails should also be introduced into the data to build a more robust model capable of operating in a modern environment.

We also found inconsistencies in model performances across various runs of the script. Statistically, this makes sense as different folds in the cross-validation methodology will naturally have some variance within them. Re-running the models with a higher number of folds and adding more parameters to the Grid Search will ensure that the best model could be correctly selected for clients with the computing resources for more powerful solutions. Additional testing for larger clients may include experimenting with various deep learning architectures to better classify spam emails.