TF-IDF for Ham/Spam Classification

Sovann Chang
Vanderbilt University
sovann.d.chang@vanderbilt.edu

ABSTRACT

In this paper, we examine the effectiveness of using the natural language processing (NLP) method term frequency-inverse document frequency (TF-IDF) to classify emails as spam (junk) or ham (legitimate). Our methodology first involves using the spaCy package to get a tokenized representation of the texts. There are many misspelled or otherwise illegitimate words contained within the emails, some of which may be due to low-quality spam or the result of an attempt to convert visual aspects of an email into text. These may cause problems with spaCy's tokenization, so we remove these invalid words, keeping track of the number contained in each document. We then create a TF-IDF matrix ranking the values of each word in each document, excluding stop words. We then run a logistic regression model on the resulting matrix, training it to accurately predict the email's label of ham or spam. The findings reflect the potential to detect spam emails using only TF-IDF. In addition to this regression model, we train a second model on the same data, but add an extra feature representing the percentage of words in each email that were invalid. We compare the performances of the two models to assess the effectiveness of considering invalid word percentage when classifying emails.

Keywords

natural language processing, tf-idf, ham, spam, email classification, logistic regression

1. INTRODUCTION

The first spam email was sent in 1978, but the development of the internet in the 1990s is when spam really began to become common [1]. 30 years later, spam has become so prevalent that our emails and messaging apps have filters to keep spam out. With the rise of generative AI models, spam and phishing emails seem set to become even more common and more difficult to identify [2]. With this in mind, distinguishing spam emails from legitimate (or "ham") emails will be even more important, and our methods for doing so will need to be sound.

There are many ways for humans to visually identify spam, but most of them are easily countered by spammers who ensure that their emails look as normal as possible. When emails are passable visibly, the text must be analyzed. Any good automatic spam filter needs to parse the text of an email to determine its validity. It must learn to identify patterns in the text that are indicative of spam. The simplest form of this is to identify words that are often used in spam emails but rarely used in legitimate emails.

Term frequency-inverse document frequency provides an estimation of this intangible quality. TF-IDF is a combination of two different measurements. Term Frequency represents how often a term appears in a document, calculated as the number of times a term appears divided by the number of terms in the document. Document Frequency is a representation of how common a term is in an entire corpus, calculated by dividing the number of documents in which the term appears by the total number of documents in the corpus. *Inverse* Document Frequency is, in a sense, its opposite. It can be thought of as "How unique is this term to this document?" and is calculated by taking the reciprocal (numerator and denominator swap places) of Document Frequency and taking the natural log of the result. The Term Frequency and Inverse Document Frequency are then multiplied to yield TF-IDF.

Conceptually, this calculation produces a combination of "How important is this word to this document?" and "How unique is this word to this document?". If a word is both fairly important and relatively unique to multiple spam documents, it is a good indicator of spam. For this reason, TF-IDF is a common method for spam detection [3, 4, 5].

2. PURPOSE STATEMENT AND RESEARCH QUESTIONS

The goal of this study is to investigate the following research questions:

1. Can a logistic regression model be trained using only TF-IDF data to accurately distinguish spam emails from ham emails?
2. Does including the rate of misspelled or otherwise invalid words in an email provide a noticeable (2% or greater) increase in the model's performance?

The first research question is important to consider because TF-IDF was the traditional method for email classification before the rise of artificial intelligence, which is capable of detecting spam with a high success rate [6]. Logistic regression is one of the most basic machine learning algorithms and TF-IDF is a reasonably simple value to calculate. If such a simple algorithm can classify emails using only TF-IDF, this reflects very well on the robustness of TF-IDF as a metric for identifying spam. Additionally, if this combination of a simple statistic and basic algorithm is successful, then most technologically literate people could feasibly create their own spam detectors.

The second research question arose after some data exploration. When calculating term frequency, we discovered that some of the most common terms that appeared in the emails were not English words, acronyms, or common shorthand. The "Methods" section elaborates more on these terms. Spelling errors have historically been indicative of spam emails, so we decided to investigate the extent to which invalid words could be used to improve our model.

3. LITERATURE REVIEW

3.1 Other TF-IDF Methods

Term frequency-inverse document frequency has long been a useful tool for spam detection. There are published papers as far back as 2004 [8, 9] that discuss using TF-IDF alongside algorithms such as Naïve Bayes Classifiers (NBCs) and Support Vector Machines (SVMs) to classify emails as ham or spam. These are the most common methods usually used with TF-IDF, and when trained by experts, can yield near-perfect accuracies. Some examples of this are an NBC trained by Jaswal, Das, and Khushboo [3], which has a test accuracy of 99% and a comprehensive study of different SVM kernels by Amayri and Bouguila [10] that show F1 scores as high as 85% on TF-IDF data.

Although Naïve Bayes and SVM are slightly more common, some researchers have used TF-IDF data to train Random Forest algorithms [11] to achieve 97.5% accuracy and clustering algorithms [12] such as K-Means and BIRCH to achieve accuracies above 90%.

Finally, there are those who have done the same thing that we set out to achieve in this paper. Park, Qureshi, and Shin [13] have trained a logistic regression model on TF-IDF data and achieved a test accuracy of nearly 95%. However, this does not render our efforts useless. While it can be beneficial to study the work of others, hands-on experience in testing a hypothesis is still crucial to the understanding of a subject.

3.2 Methods not involving TF-IDF

While TF-IDF is a common measure to use in algorithms for spam detection, there are still many studies that elect not to use it. Carreras and Màrquez [14] utilized AdaBoost to classify emails without using TF-IDF, achieving accuracy scores above 90%.

One particularly interesting work on spam detection was written by Stuart, Cha, and Tappert [15]. They identify some flaws in the traditional Naïve Bayes approach, stating that NB approaches “rely upon a consistent vocabulary by the spammers,” [15] and “do not identify the common patterns in spam that humans can easily and readily identify, such as unusual spellings, images and hyperlinks” [15]. For these reasons, they train a neural network with greater than 90% accuracy on various features from an email’s subject header, priority and content-type headers, and body. These features focus on specific characteristics of terms (such as length, specific letters contained, case, and hyperlink status) rather than which terms appear most often.

Google, the company that owns one of the most popular email platforms in Gmail, developed their own text vectorizer called RETVec [6]. RETVec combines “a novel, highly-compact character encoder, an augmentation-driven training regime, and the use of metric learning.” [6] to improve Google’s spam detection algorithm by 38%. It is now used in Gmail’s spam filter [6].

4. METHODS

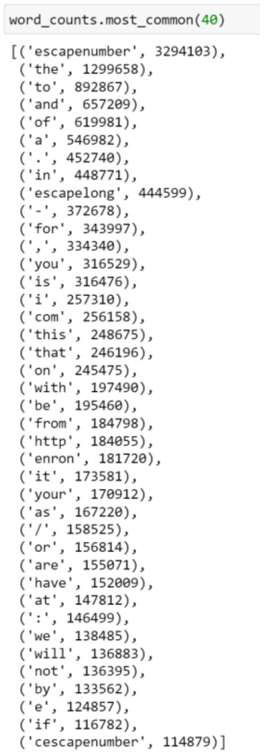
4.1 Dataset

The sample emails being used for this analysis are taken from a Kaggle dataset [7] containing nearly 194,000 total ham and spam emails. The dataset is relatively clean and simple, containing just two variables: the text of the email and the spam/ham label. The author describes the dataset as a combination of multiple other datasets, although they do not list the other datasets. While we acknowledge that it is plausible that this data is substandard, the content and format of emails vary widely, so most samples of emails can still be useful, even if they are not wholly representative of all emails.

Due to the size of the dataset and the time required to run spaCy’s nlp() function on the text of the emails, a random sample (seed 42) of 20,000 emails was taken as a trimmed dataset. After the data cleaning and feature engineering, the data was given a random (seed 42) 80/20 train/test split. The result was a training dataset with 16,000 emails and a testing dataset with 4,000 emails.

4.2 Data Cleaning

As mentioned previously, many of the emails contained terms that were not recognizable as English words, acronyms, or common shorthand. The 40 most common terms in the emails are shown in the figure below.



```
word_counts.most_common(40)
```

```
[('escapenumber', 3294103),
 ('the', 1299658),
 ('to', 892867),
 ('and', 657209),
 ('of', 619981),
 ('a', 546982),
 ('.', 452740),
 ('in', 448771),
 ('escapelong', 444599),
 ('-', 372678),
 ('for', 343997),
 (',', 334340),
 ('you', 316529),
 ('is', 316476),
 ('i', 257310),
 ('com', 256158),
 ('this', 248675),
 ('that', 246196),
 ('on', 245475),
 ('with', 197490),
 ('be', 195460),
 ('from', 184798),
 ('http', 184055),
 ('enron', 181720),
 ('it', 173581),
 ('your', 170912),
 ('as', 167220),
 ('/', 158525),
 ('or', 156814),
 ('are', 155071),
 ('have', 152009),
 ('at', 147812),
 (':', 146499),
 ('we', 138485),
 ('will', 136883),
 ('not', 136395),
 ('by', 133562),
 ('e', 124857),
 ('if', 116782),
 ('cescapenumber', 114879)]
```

Figure 1. 40 most common terms in the emails before cleaning

Clearly, some very common "words" are clearly not actual words. "escapenumber" is nearly three times as common as any other term, and "escapelong" and "cescapenumber" also are present in the top 40. In order to filter out these bad terms, we turned to the nltk library. nltk contains two subpackages that helped us with this problem. The first, wordnet, was created by Princeton, and is a large lexical database of English nouns, verbs, adjectives, and adverbs [INSERT NUMBER]. The second, nltk.corpus, contains "words", a standard dictionary of words recognized by Unix systems [INSERT NUMBER]. The reason that both of these dictionaries were used is that there are words that should be considered valid but are only contained within one of them. For example, “bible” is only considered a word by wordnet and “for” is only contained within nltk.corpus.words because it is not one of the parts of speech covered in wordnet. If a term was not contained within at least one of the dictionaries, it was considered “invalid” and removed from the text of the email.

In accordance with the second research question, a count of the number of invalid terms within each email was kept, along with a count of the total terms in the text. These two numbers were used to obtain the percentage of invalid terms in each email.

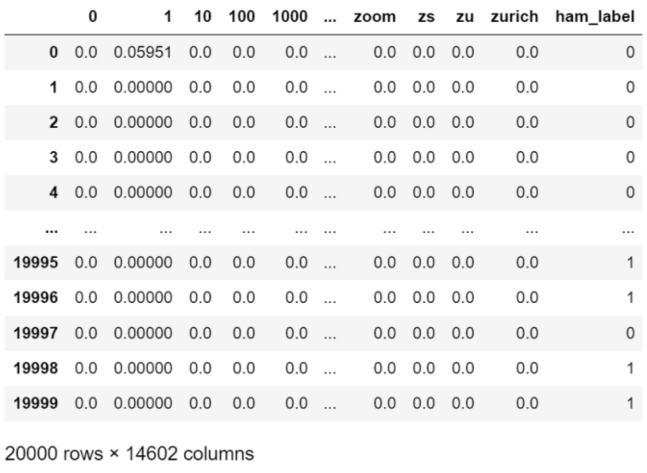
The last step in cleaning the data occurred during the feature engineering and will be discussed below.

4.3 Feature Engineering

The first step of the feature engineering was to tokenize the text of each email. We passed the text of all 20,000 emails, with invalid words removed, into spaCy's nlp.pipe() function. After nearly 20 minutes, the function returned each email's text as a list of tokens, ready to be used for a TF-IDF calculation.

To perform TF-IDF, we utilized the TfidfVectorizer function from the scikit-learn package. This function allows the user to manually define a tokenizer to alter each input document. We defined a tokenizer that removed whitespace tokens, punctuation tokens, and stop words. Stop words are words that are commonly used in English, but do not provide much information. Examples include determiners like "the" and "a" as well as prepositions like "his" and "her". They are removed from text before performing many NLP tasks because they can affect the outcome in ways that do not reflect reality. For example, if two texts are analyzed for similarity and both texts contain many of these stop words, they will likely be considered similar simply because they both use words like "the" or "his", even if the remainder of the content of the texts has nothing in common. Removing stop words is a standard practice in NLP [16].

The TfidfVectorizer function also supports a parameter called min_df. Setting a min_df value returns only terms that appear in at least that many documents within the corpus. For the purposes of this experiment, we set min_df equal to 5. The function returned a pandas dataframe containing 20,000 rows (each representing one document) and 14,601 columns (each representing one term that appears in at least 5 documents). We then set each column's name to be the term it represents. Finally, we added a column representing the ground truth label to this dataframe. A sample of the final dataframe is shown below.



	0	1	10	100	1000	...	zoom	zs	zu	zurich	ham_label
0	0.0	0.05951	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0
1	0.0	0.00000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0
2	0.0	0.00000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0
3	0.0	0.00000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0
4	0.0	0.00000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0
...	...	...	...	...	...	...	...	...	...	...	...
19995	0.0	0.00000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	1
19996	0.0	0.00000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	1
19997	0.0	0.00000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0
19998	0.0	0.00000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	1
19999	0.0	0.00000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	1

20000 rows × 14602 columns

Figure 2. TF-IDF dataframe with labels

The above sample of the dataframe shows almost entirely zero values for most term/document combinations, meaning that most documents shown do not contain the words shown. However, document 0 does have a nonzero value for the term "1", indicating that "1" appears in document 0 and has a TF-IDF value of 0.05951 for it.

4.4 Modeling

After splitting the data into the 80/20 train/test sets, we used a scikit-learn logistic regression model with default parameters to process the TF-IDF data. Because this was an exploratory, educational project, we did not explore any alternative hyperparameters such as a different loss, tolerance, or maximum number of iterations.

We believe that logistic regression is an appropriate model to use for the data. One reason for this is that TF-IDF data is inherently sparse, which can cause problems for some models. The default scikit-learn logistic regression model applies regularization [17], which limits the risk of overfitting by reducing the data's complexity [18].

Another advantage of using logistic regression is that the model inherently learns the weights of each feature. These feature coefficients are directly correlated with how important the feature is to the prediction. Not only can we extract a ranking of how important each term is, but the weights are also signed, meaning that the most negative coefficients are heavily predictive of ham, while the most positive coefficients are heavily predictive of spam. The ability to specifically identify which terms are indicative of ham and spam emails is extremely valuable.

5. RESULTS

5.1 Model without Invalid Word Rate

The logistic regression model that did not include invalid word percentage as a parameter performed quite well. Out of the 4,000 total documents in the test set, 2,086 were ham emails and 1,914 were spam emails. Of the 2,086 true ham emails, 1,960 were correctly classified as ham and 126 were incorrectly predicted as spam. Of the 1,914 true spam emails, 1,836 were correctly classified as spam and 78 were incorrectly predicted as ham. These results are shown in the confusion matrix below.

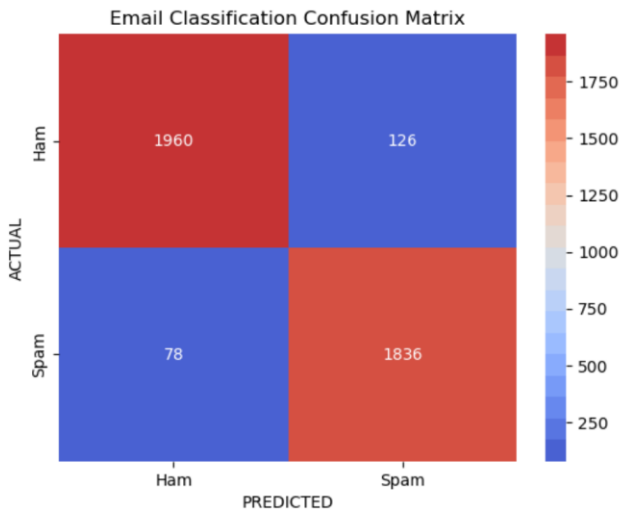


Figure 3. Confusion matrix for logistic regression model not including invalid word percentages

We look at four metrics for evaluating the performance of the model. All four scores are shown in the table on the next page.

Table 1: Evaluation metrics for logistic regression model not including invalid word percentages

Accuracy	Precision	Recall	F1 Score
0.9490	0.9358	0.9592	0.9474

Our first metric for evaluating the model is pure accuracy, calculated as the number of correctly classified documents divided by the total number of documents. The model's accuracy was 94.9%.

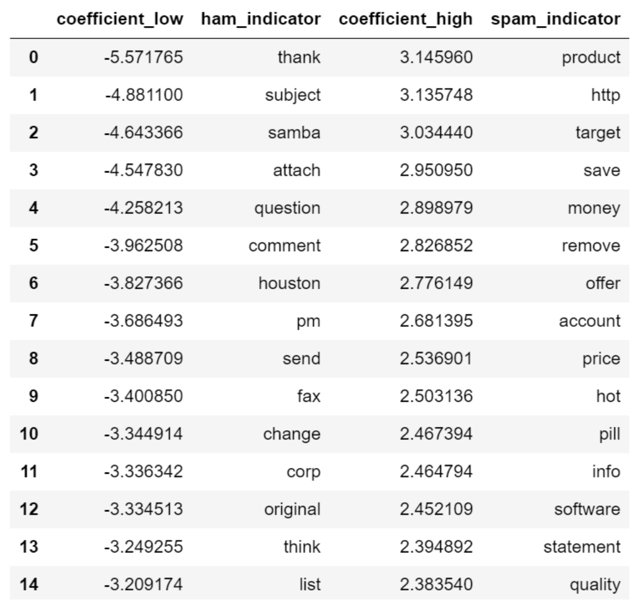
The second result is precision. Precision can be thought of as "when we predict spam, how accurate are we?" It is calculated by dividing the number of True Positives (predicted spam AND actually is spam) by the number of predicted positives (predicted spam regardless of actual value). The model's precision was 93.58%.

The third result is recall. Recall can be thought of as "when we see spam, how good are we at identifying it?" It is calculated by dividing the number of True Positives (predicted spam AND actually is spam) by the number of actual positives (actually is spam regardless of prediction). The model's recall was 95.92%.

The fourth result is F1 score. F1 score is a metric that is designed to balance the precision and recall scores. It is calculated using the equation $2 * (\frac{precision * recall}{precision + recall})$, which is the harmonic mean of precision and recall. F1 score is generally considered to provide a comprehensive evaluation of a model. The model's F1 score was 94.74%.

All four of these metrics were significantly above 90%. Because our classes were well-balanced (~52% ham, ~48% spam), this performance is exceptional for such a simple model.

The logistic regression model allows us to identify the most important terms used to identify ham and spam emails, shown below.



	coefficient_low	ham_indicator	coefficient_high	spam_indicator
0	-5.571765	thank	3.145960	product
1	-4.881100	subject	3.135748	http
2	-4.643366	samba	3.034440	target
3	-4.547830	attach	2.950950	save
4	-4.258213	question	2.898979	money
5	-3.962508	comment	2.826852	remove
6	-3.827366	houston	2.776149	offer
7	-3.686493	pm	2.681395	account
8	-3.488709	send	2.536901	price
9	-3.400850	fax	2.503136	hot
10	-3.344914	change	2.467394	pill
11	-3.336342	corp	2.464794	info
12	-3.334513	original	2.452109	software
13	-3.249255	think	2.394892	statement
14	-3.209174	list	2.383540	quality

Figure 4. The 15 most impactful terms for identifying spam and ham emails in the model without invalid word rates

While the words that are indicative of ham emails are not particularly notable, many of the words that are indicative of spam emails make sense. Common spam emails will try to sell people a *product*, advertising *pills* or *software*, sending them links with *http* in them, and promising that their targets will *save money* if they just create an *account* to accept the spammers' *offer*. It is hardly any surprise that these terms are signs of spam – these are keywords that humans look out for when we try to identify spam.

5.2 Model with Invalid Word Rate

The logistic regression model that included invalid word percentage as a parameter performed almost exactly the same as the model without it. The same data was used for both, so the percentage of true ham and true spam emails is the same. This model correctly classified 1,968 of the 2,086 true ham emails and 1,829 of the 1,914 true spam emails. This means that 118 ham emails were predicted as spam and 78 spam emails were predicted as ham. These results are shown in the confusion matrix below.

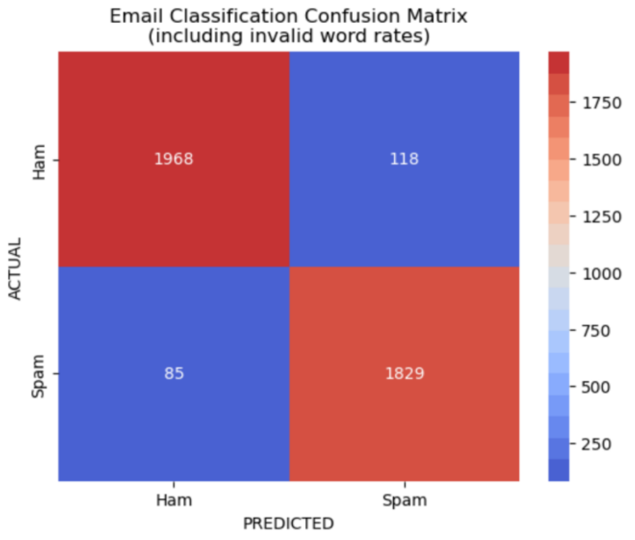


Figure 5. Confusion matrix for logistic regression model including invalid word percentages

The model that included the percentage of words in each email as a feature correctly classified 8 more ham emails, but incorrectly predicted ham on 7 more spam emails than the model with just the TF-IDF values.

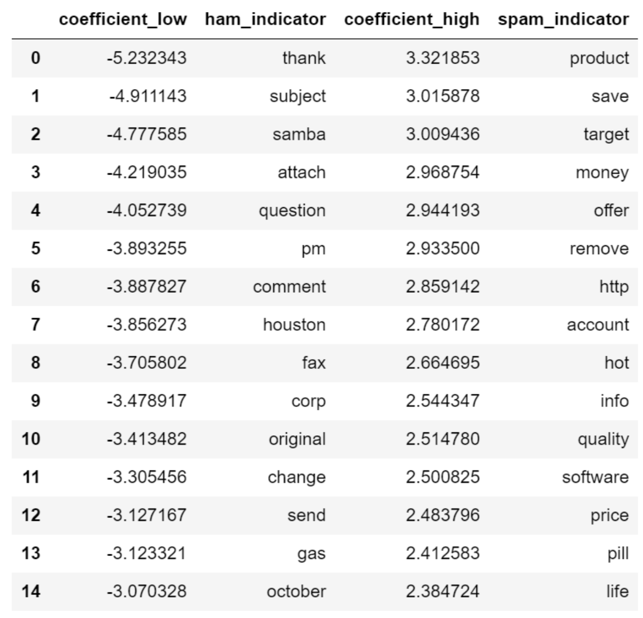
We compare the performance of the model with the invalid word percentages to the model without invalid word percentages.

Table 2: Evaluation metrics for both logistic regression models

Model	Accuracy	Precision	Recall	F1 Score
Without invalid	0.9490	0.9358	0.9592	0.9474
With Invalid	0.9493	0.9394	0.9556	0.9474

All four of the accuracy scores yielded nearly the same value (within 1%) for both models. In fact, the F1 scores, perhaps the most balanced estimations of accuracy. were within 0.01% (they appear the same due to rounding)

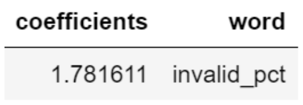
Again, we want to examine the most important terms for identifying ham and spam emails, shown in the figure on the following page.



	coefficient_low	ham_indicator	coefficient_high	spam_indicator
0	-5.232343	thank	3.321853	product
1	-4.911143	subject	3.015878	save
2	-4.777585	samba	3.009436	target
3	-4.219035	attach	2.968754	money
4	-4.052739	question	2.944193	offer
5	-3.893255	pm	2.933500	remove
6	-3.887827	comment	2.859142	http
7	-3.856273	houston	2.780172	account
8	-3.705802	fax	2.664695	hot
9	-3.478917	corp	2.544347	info
10	-3.413482	original	2.514780	quality
11	-3.305456	change	2.500825	software
12	-3.127167	send	2.483796	price
13	-3.123321	gas	2.412583	pill
14	-3.070328	october	2.384724	life

Figure 6. The 15 most impactful terms for identifying spam and ham emails in the model that includes invalid word rates

The terms present in this model are almost all the same as the top 15 terms from the other model, just in a slightly different order. This is expected since the models have 14,601 of the same features and only one different feature. We do note that the invalid word count does not appear in the list of the most important features. We are interested in knowing its importance to the model's predictions.



coefficients	word
1.781611	invalid_pct

Figure 7. The importance of the feature invalid_pct

We observe that the percentage of invalid words is positive 1.7816, showing that it is indicative of spam to a moderately high degree.

6. CONCLUSIONS AND DISCUSSION

The purpose of this paper was to investigate the effectiveness of a logistic regression model trained only on Term frequency-inverse document frequency data. A secondary research question was developed after discovering a high number of terms that are not English words, acronyms, or shorthand. It was theorized that using the percentage of these invalid words could improve the performance of the model.

Regarding the first research question, our findings clearly indicate that a good logistic regression model can be trained only using TF-IDF data. Given that our testing data had a ground truth ratio of around 52% ham and 48% spam, a model with an F1 score of nearly 95% is an extremely successful one. That a simple logistic regression model could perform so well shows that TF-IDF is a robust method of generating data that can be used to classify emails as ham or spam.

While the first research question focused on an established method, the second research question addressed something more novel. None of the literature referenced in this paper examined the quantity of linguistically problematic words. While the model did use invalid word percentage as a moderately strong indicator of spam, its performance was almost exactly the same as the model that did not incorporate the extra data, to our surprise. There are many potential reasons that this did not increase the accuracy of the model. There is no universally agreed-upon list of valid words that can be used in an email, so our approximation of this certainly counted some terms as invalid that were not. It is also possible that the nature of many of the invalid terms is not correlated with the difference between ham and spam emails. In particular, three of the most common invalid terms were "escapenumber", "escapelong" and "cescapenumber", which were almost certainly not written in the original emails. Rather, it is likely that in the conversion from multimedia email to text only, some elements of the email were not easily converted to text, and this was the chosen representation. This may be equally likely to occur in ham emails and spam emails, leading to noise that affects the usefulness of the variable.

Overall, we feel that the experiments were a success. We addressed both of our research questions using sound methods and produced satisfying results. We provided further evidence for the quality of TF-IDF as a metric and investigated the usefulness of a new attribute of an email. We hope that the insights presented in this paper will be valuable to those wishing to innovate within the realm of spam detection.

7. REFERENCES

The GitHub repository containing this work can be found at https://github.com/SovannChang/nlp-final

[1] Britannica, T. Editors of Encyclopaedia (2023, December 19). *spam. Encyclopedia Britannica*. https://www.britannica.com/topic/spam

[2] Violino, B. (2023, November 28). *AI tools such as CHATGPT are generating a mammoth increase in malicious phishing emails*. CNBC. https://www.cnbc.com/2023/11/28/ai-like-chatgpt-is-creating-huge-increase-in-malicious-phishing-email.html

[3] Jaiswal, M., Das, S., & Khushboo. (2021). Detecting spam e-mails using stop word tf-IDF and stemming algorithm with naïve Bayes classifier on the multicore GPU. *International Journal of Electrical and Computer Engineering (IJECE), 11*(4), 3168. https://doi.org/10.11591/ijece.v11i4.pp3168-3175

[4] Hossain, S.M.M.; Kamal, K.M.A.; Sen, A.; Sarker, I.H. TF-IDF Feature-Based Spam Filtering of Mobile SMS Using Machine Learning Approach. *Preprints* 2021, 2021090251. https://doi.org/10.20944/preprints202109.0251.v1

[5] Chavez, Alan (2020) *TF-IDF classification based Multinomial Naïve Bayes model for spam filtering*. Masters thesis, Dublin, National College of Ireland. https://norma.ncirl.ie/4490/

[6] Bursztein, E., & Zhang, M. (2023, November 29). *Improving text classification resilience and efficiency with retvec*. Google Online Security Blog. https://security.googleblog.com/2023/11/improving-text-classification.html

[7] Likith, M. (2024, March 14). *190K+ SPAM: Ham Email dataset for Classification*. Kaggle. https://www.kaggle.com/datasets/meruvulikith/190k-spam-ham-email-dataset-for-classification/data

[8] R. Matsumoto, Du Zhang and Meiliu Lu, "Some empirical results on two spam detection methods," *Proceedings of the 2004 IEEE International Conference on Information Reuse*

and Integration, 2004. IRI *2004*., Las Vegas, NV, USA, 2004, pp. 198-203, doi: 10.1109/IRI.2004.1431460.

[9] Chih-Chin Lai and Ming-Chi Tsai, "An empirical performance comparison of machine learning methods for spam e-mail categorization," *Fourth International Conference on Hybrid Intelligent Systems* (HIS'04), Kitakyushu, Japan, 2004, pp. 44-48, doi: 10.1109/ICHIS.2004.21.

[10] Amayri, O., Bouguila, N. A study of spam filtering using support vector machines. *Artif Intell Rev* 34, 73–108 (2010). https://doi.org/10.1007/s10462-010-9166-x

[11] Nilam Nur Amir Sjarif, Nurulhuda Firdaus Mohd Azmi, Suriayati Chuprat, Haslina Md Sarkan, Yazriwati Yahya, Suriani Mohd Sam. SMS Spam Message Detection using Term Frequency-Inverse Document Frequency and Random Forest Algorithm. *Procedia Computer Science, Volume 161*, 2019, Pages 509-515, ISSN 1877-0509. https://doi.org/10.1016/j.procs.2019.11.150. (https://www.sciencedirect.com/science/article/pii/S1877050919318617)

[12] Basavaraju, M., & Prabhakar, Dr. R. (2010). A novel method of spam mail detection using text based clustering approach. *International Journal of Computer Applications*, 5(4), 15–25. https://doi.org/10.5120/906-1283

[13] Al-Besher, A., Kumar, K., Sangeetha, M., & Butsa, T. (2021). Pseudo NLP joint spam classification technique for Big Data Cluster. *Computers, Materials &amp; Continua*, 71(1), 517–535. https://doi.org/10.32604/cmc.2022.021421

[14] Carreras, X., Marquez, L (2001). Boosting Trees for Anti-Spam Email Filtering. *Proceedings of RANLP-2001*, pp. 58-64, Bulgaria, 2001. https://arxiv.org/abs/cs/0109015. https://doi.org/10.48550/arXiv.cs/0109015

[15] Stuart, I., Cha, SH., Tappert, C. (2004). A Neural Network Classifier for Junk E-Mail. In: Marinai, S., Dengel, A.R. (eds) Document Analysis Systems VI. DAS 2004. *Lecture Notes in Computer Science, vol 3163*. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-28640-0_42

[16] Ganesan, K. (2023, March 16). *What are stop words?*. Kavita Ganesan, PhD. https://kavita-ganesan.com/what-are-stop-words/

[17] *Sklearn.linear_model.logisticregression*. scikit. (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[18] Rosidi, N. (2023, April 7). Best machine learning model for sparse data. KDnuggets. https://www.kdnuggets.com/2023/04/best-machine-learning-model-sparse-data.html