Ham Spam Filtering
Classifying Emails by Text

Jeremy Randolph
Fundamentals of Analytics and Discovery Informatics
12/13/2020

Introduction

Electronic Mail or Email is the method of exchanging messages between people using electronic devices. Ray Tomlinson invented email in 1971 as a means of encrypted communication between host computers. Email quickly became a popular means of communication by businesses and individuals and has seen widespread use ever since. Companies like Apple, Microsoft, Yahoo, and Google now make up the majority of the market share of email service providers.

Today there are about 3.9 billion daily email users as well as over 5.6 billion active email accounts. [1] Approximately 281 billion emails are sent and received every day with that number to rise significantly in the coming years. [3] With free-flowing communication across the world now possible for little to no cost, many companies have capitalized on this technology to perform mass advertising campaigns. While others who are looking to scam individuals or infect computer systems for personal gain gained a valuable asset to help further their plans.

Spam can be categorized as unwanted and bulk mail aimed at either inundating email users with advertisements or more so scamming users or infecting systems with malicious software. In March 2020 alone spam accounted for 53.95% of all email traffic. [2]. While that number seems high in 2012 spam accounted for 69% [2]. Email Service Providers have had to develop solutions for this problem not just for users but for themselves. Spam email can take up a large portion of a computer's network, hardware, and storage capacities costing companies efficiency and money.

Early applications of rule-based filtering focused on features of a spam email and performed well enough catching 79.7% of spam mail with only a 1.2% false-positive rate. [8] However, as accuracy improves the likelihood of a non-spam email being miscategorized increases. This could pose a massive problem for users as important mail not seen is significantly worse than spam mail seen.

Today email service providers use an array of applications to catch spam with unheard-of accuracies. The development of statistical algorithms, as well as more complex neural networks, has reduced the number of spam emails immensely. With the application of machine learning, Google has increased the accuracy and precision of spam classification to 99.97%. [4]

Background

The objective of this project was to evaluate the three more popular machine learning algorithms for their effectiveness in email spam classification. Naive Bayes, Support Vector Machine, and a Neural Network was implemented for the task. The implementation of these algorithms was aimed to attempt to attain similar results described in the literature review. Several key metrics were used to measure the performance of each of these algorithms.

Naive Bayes was the simplest of the three algorithms chosen and the easiest to implement. Naive Bayes can be described as a simple technique used for instance classification and can be trained very well on supervised data, or data that contains labels for each instance or the output of every instance is known. Naive Bayes assumes that the value of a particular feature is independent of any other feature. Although quite simple Naive Bayes performs quite well in many real-world situations and often requires only a small number of training instances.

Support Vector Machine is a supervised learning algorithm built upon the statistical framework of VC theory. The model is a representation of a set of instances mapped as points within space. Lines are drawn so that the mapped instances can be divided by gaps as wide as possible. New data when added are predicted to belong to a category based on which side of the gap they are mapped to. Support Vector Machines perform quite well with high dimensional data.

Lastly, an Artificial Neural network is a computing system inspired by the neural networks that constitute biological brains. A collection of nodes are assembled with each node having the ability to transmit a signal like a neuron in the brain. Weights are applied to each node in the process to change the signal strength. Data is fed into the network to be trained to form probability-weighted associations between inputs and output which can be used to predict future data. Each of these three algorithms was evaluated and optimized to perform email classification

Literature Review

In preparation for this project, a literature review was completed. Several articles were identified and useful in guiding the to the current point. Three articles, in particular, “Machine learning for email spam filtering: review, approaches, and open research problems”,[4] ” Analysis of Machine Learning Algorithms for Email Classification Using NLP”,[5] and “A novel hybrid approach of SVM combined with NLP and probabilistic neural network for email phishing”,[9] will be discussed below in detail.

The first of the entries to be discussed is the article “Machine learning for email spam filtering: review, approaches, and open research problems”.[4] This study looked into most modern-day supervised learning algorithms and evaluated the efficacy of all of them regarding email spam. The article outlined the necessary steps needed to prepare data for modeling. The process was listed as follows preprocessing, tokenization, and feature selection. During the preprocessing steps, the emphasis was placed on dividing the body of each email into its meaningful parts and removal of all filler information. Large email text was compressed into feature vectors. Regarding evaluation metrics, the total cost ratio should be used the asses performance of machine learning models as false positives, and categorizing an important email as spam should weigh heavily on the fitness of a given model. For each supervised learning technique, a pseudocode algorithm was listed to describe the processes involved For each proposed algorithm the limitations were evaluated. During the evaluation, computer resources need to be taken into consideration when implementing neural networks. The paper offered a perfect starting point for further research and guided the choice of algorithms and evaluation metrics

The Second entry to be discussed is the article ” Analysis of Machine Learning Algorithms for Email Classification Using NLP”[5]. While not as detailed and simplistic with algorithm choice as the first entry this article went more in-depth around pre-processing and actual evaluation of data. During preprocessing the article extracts URLs, phone numbers, and addresses and replaces them with terms like “ladder” to describe an email address. The author asserts this will increase the accuracy of the models. The steps used in the articles preprocessing will play a key role in future work. To prepare the data for modeling lowercasing, normalizing the words, word stemming, removal of non-words and the removal of stop-words were used. The results concluded from the paper will help drive the future analysis of this project. To highlight, the Naive Bayes algorithm tested performed rather poorly and succumbed to overfitting. However, the Neural Network algorithm used as well as the SVM algorithm performed exceptionally well and scored a ninety percent pull on the test data. This will provide a great reference later.

The final entry to be discussed is the article “A novel hybrid approach of SVM combined with NLP and probabilistic neural network for email phishing”.[9] The focus of the paper was spent on classifying ham, spam, and phishing. The key differences between the three categories are ham is requested and genuine mail, spam is unsolicited and spontaneous mail, and phishing is mail that is unsafe and potentially malicious. Of all unwanted mail, phishing emails aim to steal user information or implant harmful software onto a user’s network or system. The author’s proposed algorithm combined a support vector machine learning algorithm and a probabilistic neural network to enhance the SVM. Any data that was missed in the first classifier was picked up in the second. Thur the use of the hybrid algorithm an increase in performance was noted versus if they were separate. The algorithm scored 98% across all metrics compared to SVM and NN which scored 87% and 90%.

The results of the literature review guided the selection of the proposed algorithms for the project. The pre-processing steps outlined will be instrumental in making sure the algorithm runs smoothly. When scoring the model extra attention will be placed on the false positive rate, precision score, and F-Beta score of each model.

Approach

Data for the project was collected from multiple sources. A large portion of the email data was scrapped from a personal Gmail account containing around seventy thousand emails as raw data. Additional spam email data was collected from two Kaggle datasets found here and here. This data was used to bolster the number of spam instances available for the algorithms to be trained on. Rather than the data from Gmail coming in a tabular form like a .csv the data was extracted as a .mbox file. A Gmail scrapper developed by the user Benwattsjoes on Github was the starting point for the data scraping. Changes to the initial code were done to extract the correct information from each email. All the important information needed for the project was selected. After merging all three sets of data the features of the dataset consisted of the subject, email text, class label, and a binary classification used for either spam or ham. The final set of records consisted of forty thousand non-spam emails or ham and two thousand spam emails. This data was exported as a CSV for later use.

subject	text	class_labe	ham_span
NEWSLETT	["See inside for NASM's February Newsletter.\n\r	ham	0
9 games p	['App Store\nYour next favorite game\nLooking fo	ham	0
=?utf-8?B	["Perfect 10 delivery\nEmail not displaying prope	ham	0
=?UTF-	["\n\n\nOffice Depot(R)Monumental Savings You	ham	0
	Subject: if you or someone you love suffers from	spam	1
	Subject: looking for a new date	spam	1
40% off at	['\n\nGap Outlet\nhttp://click.email.gapfactory.c	ham	0
$10 Bonus	["Click here to view your DICK'S Sporting Goods er	ham	0
Your Pers	["\nYour Personal YouTube Digest - Jun 28, 2012\n	ham	0

Figure 1: Sample raw data set before feature extraction

A pandas data frame was used to hold the data in memory for use during the project where python was used for the rest of the project. As a part of the evaluation and to obtain the best results possible several ham to spam ratios were tested. Ratios of 1:4, 1:2, 1:1, 2:1, and 4:1. A script was created to test the ratios in the following process.

To have useful information to use several steps of data cleaning was needed. String and regex functions were applied to the data to remove all leftover HTML artifacts. All punctuations and special characters were removed. All hyperlinks, email addresses, phone numbers, or other distinct information was subbed out for generic terms such as mailId or address.

Further cleaning was done using the python package Natural Language Tool Kit or NLTK. All single-letter words and stop words were removed from the data. To further normalize the data words were lemmatized. The process in which complex words are reduced to their root word while maintaining inflection. For example, running, ran, and run all would be lemmatized to run.

The data was then split into training and testing data with a 90/10 split. Lastly, Term frequency-inverse term frequency was applied to the data so that it could be used by the algorithms. Term frequency-inverse term frequency or TF-IDF is a vectorization method to turn textual data into numerical matrix representations. The NLTK library for this was used. The process involves performing a count vectorization for the words present in the data set then the term frequency is multiplied by the inverse document frequency. This scales down the value of frequently occurring terms that are less informative.

The three algorithms were then applied to the data from the simplest to the most complex. Multinomial Naive Bayes and SVM with their appropriate kernels were developed from Scikit-Learn and the neural network was developed using Keras. Significant time was spent fine-tuning the neural network and learning rate.

The data was collected and recorded into a tabular format containing the metrics and other information from each test.

Results

	model_name	accuracy	recall	precision	f1_score	fbeta_score	ham_number	spam_number	true_neg	false_pos	false_neg	true_pos	total_average	ham_spam_ratio
0	Naive Bayes	91.92%	98.56%	91.96%	95.15%	93.21%	520	2079	33	18	3	206	94.16%	0.25
1	Svm_linear	98.46%	99.04%	99.04%	99.04%	99.04%	520	2079	49	2	2	207	98.93%	0.25
2	Svm_poly	93.08%	99.52%	92.44%	95.85%	93.78%	520	2079	34	17	1	208	94.93%	0.25
3	Svm_rbf	98.08%	99.04%	98.57%	98.81%	98.67%	520	2079	48	3	2	207	98.63%	0.25
4	Svm_sigmoid	98.46%	99.04%	99.04%	99.04%	99.04%	520	2079	49	2	2	207	98.93%	0.25
5	Neural Network	80.38%	100.00%	80.38%	89.13%	83.67%	520	2079	0	51	0	209	86.71%	0.25
0	Naive Bayes	94.23%	98.51%	92.96%	95.65%	94.02%	1040	2079	96	15	3	198	95.07%	0.5
1	Svm_linear	98.72%	99.50%	98.52%	99.01%	98.72%	1040	2079	108	3	1	200	98.89%	0.5
2	Svm_poly	96.47%	100.00%	94.81%	97.34%	95.81%	1040	2079	100	11	0	201	96.89%	0.5
3	Svm_rbf	98.08%	99.50%	97.56%	98.52%	97.94%	1040	2079	106	5	1	200	98.32%	0.5
4	Svm_sigmoid	98.40%	99.00%	98.51%	98.76%	98.61%	1040	2079	108	3	2	199	98.66%	0.5
5	Neural Network	98.40%	99.00%	98.51%	98.76%	98.61%	1040	2079	108	3	2	199	98.66%	0.5
0	Naive Bayes	96.39%	93.81%	98.38%	96.04%	97.43%	2079	2079	219	3	12	182	96.41%	1
1	Svm_linear	98.80%	98.97%	98.46%	98.71%	98.56%	2079	2079	219	3	2	192	98.70%	1
2	Svm_poly	97.84%	100.00%	95.57%	97.73%	96.42%	2079	2079	213	9	0	194	97.51%	1
3	Svm_rbf	99.04%	99.48%	98.47%	98.97%	98.67%	2079	2079	219	3	1	193	98.93%	1
4	Svm_sigmoid	98.80%	98.97%	98.46%	98.71%	98.56%	2079	2079	219	3	2	192	98.70%	1
5	Neural Network	97.84%	98.97%	96.48%	97.71%	96.97%	2079	2079	215	7	2	192	97.59%	1
0	Naive Bayes	91.19%	74.30%	100.00%	85.25%	93.53%	4158	2079	410	0	55	159	88.85%	2
1	Svm_linear	99.20%	99.07%	98.60%	98.83%	98.70%	4158	2079	407	3	2	212	98.88%	2
2	Svm_poly	99.36%	100.00%	98.17%	99.07%	98.53%	4158	2079	406	4	0	214	99.02%	2
3	Svm_rbf	99.52%	98.60%	100.00%	99.29%	99.72%	4158	2079	410	0	3	211	99.43%	2
4	Svm_sigmoid	99.04%	98.60%	98.60%	98.60%	98.60%	4158	2079	407	3	3	211	98.69%	2
5	Neural Network	99.20%	99.07%	98.60%	98.83%	98.70%	4158	2079	407	3	2	212	98.88%	2
0	Naive Bayes	90.38%	55.40%	95.93%	70.24%	83.69%	8316	2079	822	5	95	118	79.13%	4
1	Svm_linear	98.75%	98.59%	95.45%	97.00%	96.07%	8316	2079	817	10	3	210	97.17%	4
2	Svm_poly	99.04%	100.00%	95.52%	97.71%	96.38%	8316	2079	817	10	0	213	97.73%	4
3	Svm_rbf	99.62%	99.06%	99.06%	99.06%	99.06%	8316	2079	825	2	2	211	99.17%	4
4	Svm_sigmoid	98.65%	98.12%	95.43%	96.76%	95.96%	8316	2079	817	10	4	209	96.99%	4
5	Neural Network	98.65%	98.12%	95.43%	96.76%	95.96%	8316	2079	817	10	4	209	96.99%	4

Table1: Final results from the test script

The naive bayes algorithm performed well at the task with performance peaking around 98%. As being the simplest in terms of resources the naive bayes algorith could be a viable tool for email classification. However, as the number of instances in the data set increased and the ham to spam ratio increased overall model performance decreased significantly. Training the algorithm on small amounts of data would yield the best performance.

The SVM classifier used in the project exceeded expectations. While being a capable classifier in the literature it outperformed the other two models. Several Kernels were testest for their efficacy from the Sckit-Learn package in python and each yielded different results. By far the best performing model was the SVM RBF kernel with the 2:1 ham to spam ratio exceed expectations, it was able to achieve and 100% precision score and an overall score of 99.43%. With zero false positives record and only a few false negatives that SVM instance performed the best out of any other. The data needed to attain this performance was moderate and the classifier worked best with an average amount of data. A drop off in performance could be seen as the number of instances grew to 4:1.

model_name	accuracy	recall	precision	f1_score	fbeta_score	ham_number	spam_number	true_neg	false_pos	false_neg	true_pos	total_average	ham_spam_ratio
Naive Bayes	96.394%	93.814%	98.378%	96.042%	97.430%	2079	2079	219	3	12	182	0.964119331	1
Neural Network	99.199%	99.065%	98.605%	98.834%	98.696%	4158	2079	407	3	2	212	0.988799501	2
Svm_linear	98.462%	99.043%	99.043%	99.043%	99.043%	520	2079	49	2	2	207	0.989267575	0.25
Svm_poly	99.359%	100.000%	98.165%	99.074%	98.527%	4158	2079	406	4	0	214	0.990249779	2
Svm_rbf	99.519%	98.598%	100.000%	99.294%	99.716%	4158	2079	410	0	3	211	0.994255851	2
Svm_sigmoid	98.462%	99.043%	99.043%	99.043%	99.043%	520	2079	49	2	2	207	0.989267575	0.25

Figure 2: Best result for each classifier

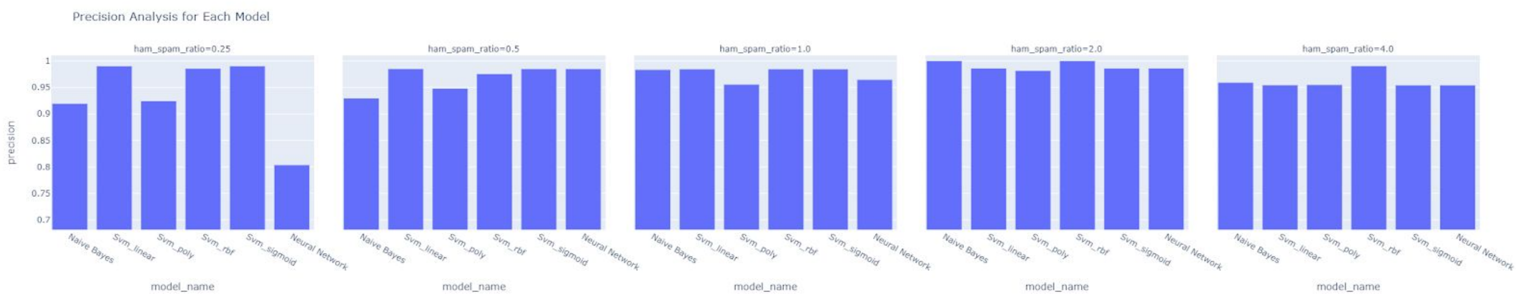


Figure 3: Precision Analysis of each model

The neural network used in the project performed about as well as descried in the literature. The model was able to attain scores above 95% with the best score record at 99%. The neural network did not perform well with limited data and struggled at the 1:4 ratio for ham to spam. However, the more information fed into the model the greater the results were. Peak performance of the model was reached around the ratio of 2:1 and 4:1 similar to the SVM models tested.

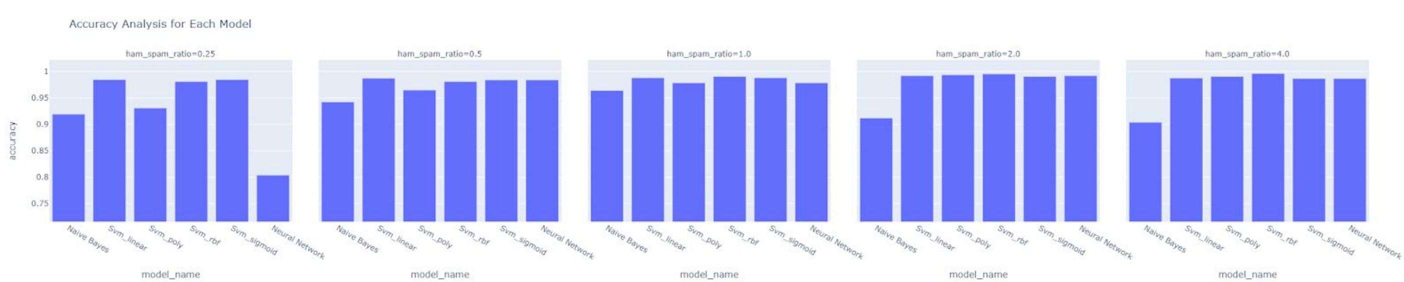


Figure 4: Accuracy of each model

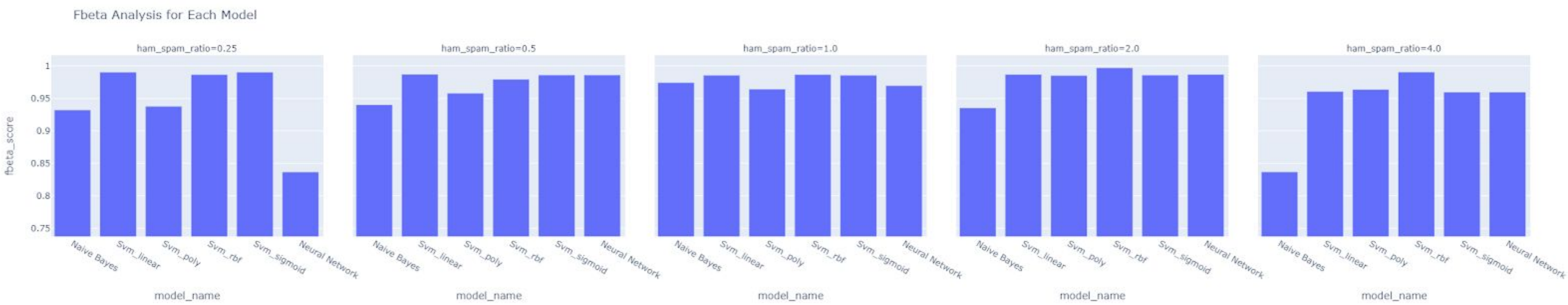


Figure 5: F-Beta Score of each model

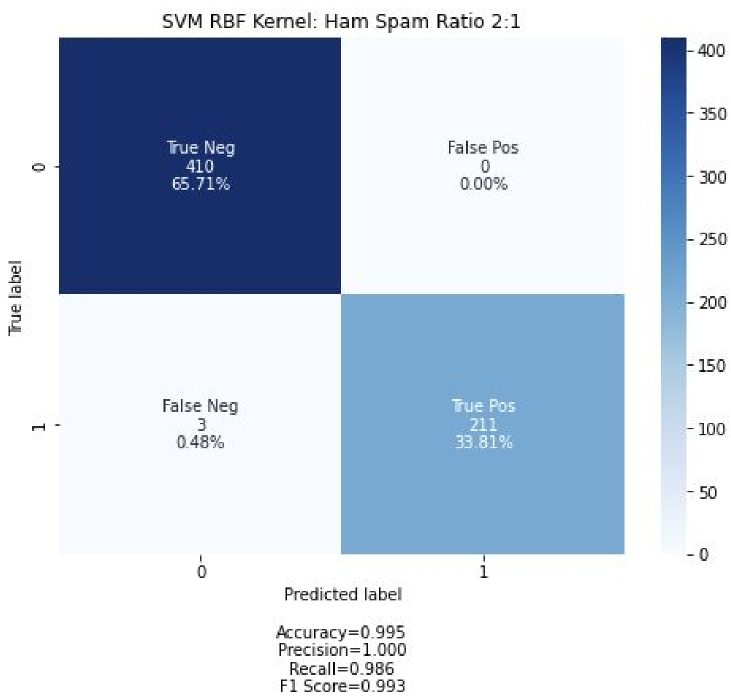


Figure 6: Confusion matrix for best performing algorithm instance

Discussion

Each of the three classifiers performed well at the task of binary classification using textual data. However, there are still several ways the performance of these models could be improved. In particular fine tuning the ham to spam ratios for each of the qualifiers could be done. The naive bayes algorithm performed the best between the ratios of 1:4 and 1:2 with a drop off around 1:1. Iterating thru different test sizes between those ratios may provide similar results. The SVM model and the Neural network performed best with higher ratios of ham to spam. The ratios of 1:1, 2:1, 4:1 yielded the best results for the two models. Again, further fine tuning and iterating thru different test sizes between the 2:1 and 4:1 may yield better results for the two classifiers.

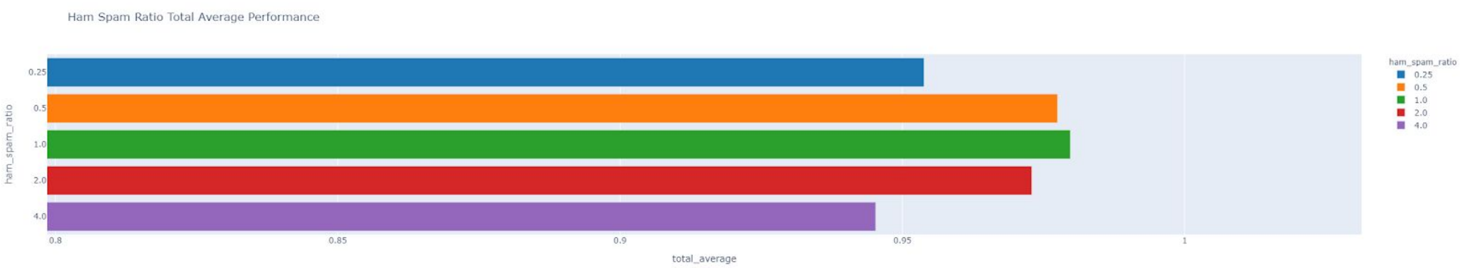


Figure 7: Ham to spam ratios on average model performance

Neural networks are complex systems of learning. Although significant time was spent fine tuning the depth, node size, and learning rate of the model used more time, energy and resources could be thrown at the algorithm. With reported neural network accuracies of around 99% there is some room for improvement.

With all three algorithms showing results over 90% and with the highest performance record at 99.43% further validation should be done to make sure the results are correct. A ten fold cross validation could be performed on the data as an extra metric. The reasoning behind this is that three different sets of data was used for the project. It should be looked into further to make sure that the models are good at classifying email spam rather than classifying which dataset the instance came from. Each of the models should be tested on the three separate datasets and their scores should be evaluated further.

Conclusion

Email today remains one of the most popular forms of communication whether person to person, business to business, or business to customer. With the larger use of email communication there will be unwanted mail sent to users whether its mass email marketing advertisements or people trying to scam or steal information from users. Over the years complex systems have been put in place to deal with that and have been overwhelmingly successful at email spam classification. Most email users do not have to worry about and inbox full of spam and never wonder about the complex operations happening in the background.

This project aimed to apply three supervised learning classification algorithms to the problem of spam classification and attempted to attain similar results described in the literature. The data was collected and aggregated from a Gmail account as well as two other datasets found on kaggle. All the information needed for the project was extracted from the raw data and formatted for latter use. A series of steps were taken to clean and prepare the data for classification. Lastly, a script was used to apply the three models to the dataset with varying ratios of ham to spam instances to train on.

All three classifiers performed well on the given task either meeting or exceeding expectations gathered from the literature review. Each of the models reached peak accuracy with different ratios of ham to spam. By far the best results attained was from a SVM RBF kernel model trained on data with a 2:1 ratio. That training model was able to score an average of 99.43% and a precision of 100% something that is incredibly important when developing classifiers for email classification.

With each model performing better at certain ratios of ham to spam further research could be done to fine tune the ratios to attain the best performance for each model. Although significant time was spent building and fine tuning the neural network more time, resources, and energy could be spent on the model to attain the best results as neural networks today could reach accuracies of 99%. Lastly, to solidify the results of this project each of the three models should be tested on each of the three datasets used for the project to confirm the classifiers indeed classify emails and not classifying which dataset the instance came from.

Extra Visualizations

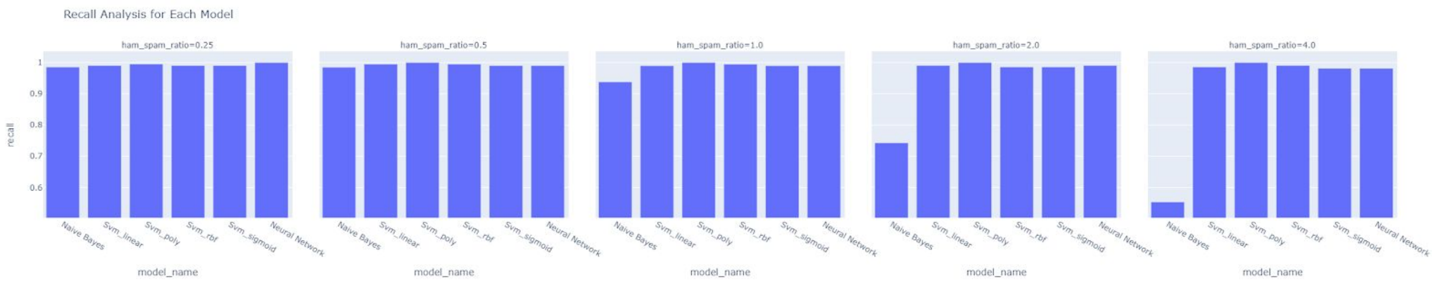


Figure 8: Recall score for each model

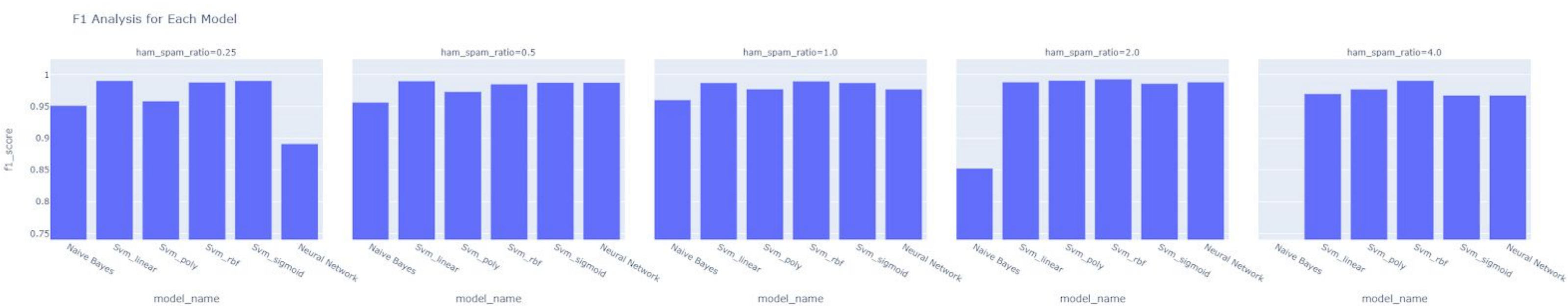


Figure 9: F-1 score for each model

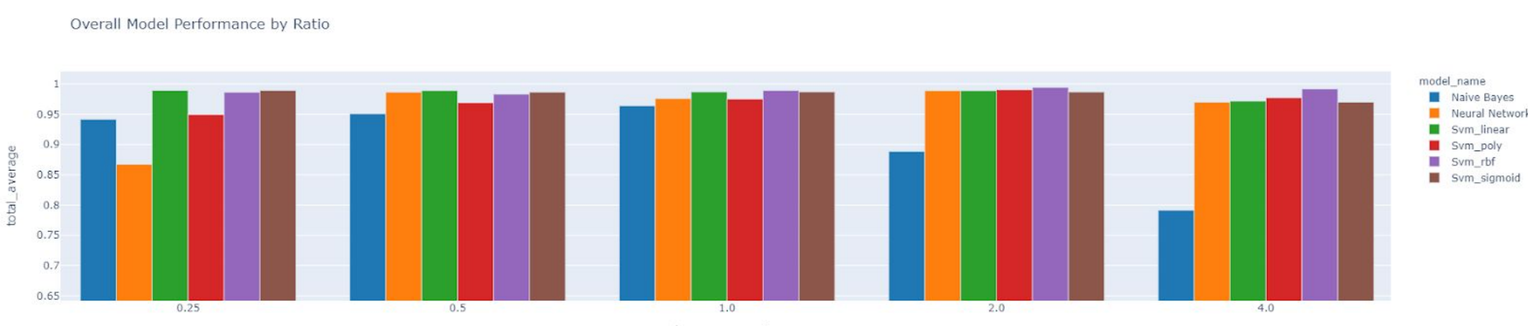


Figure 10: Overall model performance by ratio

```
data_cut.iloc[2].text
```

```
'this is not spam . you have received this e-mail because you expressed a desire in purchasing an automobile , or you recently visited one of our affiliates web sites . we apologize if this e-mail is unsolicited . please scroll down to the bottom of this message for instructions on declining ta
rgeted e-mail . + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + hello , my name is kevin cross . i have put the the auto program together to show people methods that i have used to obtain vehicles with no down payment or security deposit needed . i had bad credit , but
i did not want to drive a junker ; i wanted to drive a nice late model vehicle . i have always liked nice automobiles as long as i can remember . i have obtained my last two automobiles without having to put any money down and with my less than desirable credit . an automobile is the single big
gest expense most people make besides there home . unlike homes , most automobiles depreciate in value . that is why putting out no money on downpayments or security deposits on a car purchase or lease is smart . although the auto program was initially put together for people like myself whom h
ave bad or no credit , this program is also valuable to people with excellent or good credit , since you will be saving the money you would normally have to pay in downpayment money or security deposits . _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ + + + read what others have to say : + + + "
i ordered the auto program monday and that friday i had obtained a 1995 geo tracker 4x4 with no money down and my monthly payments are only $ 179 per month " e . b . columbus , ohio . " using the methods in your auto program i obtained a 1997 bmw convertible with no money down , thanks for show
ing me how to obtain a great car even though i had bad credit . " m . h . columbus , ohio note : ( testimonial section has full unsolicited letters from customers that have used the auto program . ) whether you have bad credit and can not get financed , or if you have excellent credit and want
to eliminate the downpayment or security deposit without increasing the monthly payment , the auto program will work for you . requirements : you must be currently employed or have a source of income and be able to make monthly payments on time . that \'s it ! no car salesman to deal with ! you
do not have to deal with the normal stress related to dealing with a car salesman . this also means you do not have to deal with getting rejected because of bad credit or having to take a car that is older and not the car you really want . _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ easy to u
se just follow the simple , step by step , directions in the auto program and obtain the late model vehicle you want . that is how easy the auto program is to use . it explains everything you need to know to obtain the late model vehicle of your choice with no downpayment or security deposit re
quired . just pay normal monthly car payments . _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ savings to you : the auto program will save you between $ 500 to $ 5000 dollars since you will not be required to put a down payment or security deposit on the vehicle you obtain . _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ no risk , 100 % money-back guarantee the 100 % money-back guarantee is in writing in the auto program . you can return the auto program anytime within 30 days for a full refund if you are not able to obtain a late model vehicle with no downpayment or security deposit .
this guarantee is regardless of your credit history as long as you use the simple methods in the auto program . _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ frequently asked questions q : how long does it take to obtain a vehicle once i receive the auto program ? a : it is all up to you and ho
w quickly you actively use the methods provided in the auto program . example : eric brown from columbus , oh , ordered the auto program on a monday and on friday of the same week he called to tell us he obtained the vehicle he wanted . q : is this program just for people with bad or no credit
? a : no , the auto program is also for people with good credit who do not want to put down the normally required downpayment . the auto program shows you how to obtain the vehicle of your choice with no money down . so , you can use your money saved from not paying a downpayment to buy better
things ! q : i can lease a car or truck from a car dealership without a down payment . . . just pay monthly lease payments . why do i need the auto program ? a : many of our customers have thought that , until they tried to lease a car from a dealership . first , you must have excellent credit
to lease a car or truck . with a lease , you have to put down a security deposit , first and last lease payments , and capitalized cost reduction which is the same thing as a down payment . . . just a different name ! when you lease a car from a dealership , you have to put down the same or mor
e as when you buy a car ! q : the monthly car payments must be high since there is no downpayment required , right ? a : the monthly car payments are the regular amount . it is just the same as if someone paid a downpayment , but through our revolutionary method , no downpayment is needed . q :
is the auto program too good to be true ? a : the step-by - step methods in the auto program have been approved by my lawyer and have been market tested by many satisfied customers . i am so confident you will benefit from the auto program that i put my name and a 100 % money-back guarantee beh
ind it ! _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ testimonials mr . kevin cross , it is with great pleasure that i give my warmest appreciation to your company for providing me with the necessary tools to successfully secure a late model chevy cavalier using the 1st and most preferred meth
od . i purchase the program on january 3 , and secured a vehicle on january 6 ; likewise , i had more than one option available to me ! i also had to opportunity to assume the lease on a late model acura legend or assume the lease on a late model bmw . but , of course , i made the wisest and mo
st financially sound decision and purchased the chevy cavalier . . . . in closing , i graciously thank you and wish you continued success in your quest to help others . professionally , d . s . + + + dear gentlemen : . . . i am writing to tell you of my good fortune in acquiring a new ford esco
rt wagon from ricart ford . i think that your program did help instill some confidence in me that i could get a new car . at least i knew that i had to do something about the piece of junk i was driving . thank you for your assistance in helping me realize my dream of owning a new vehicle . i s
till can\'t believe my good fortune . i am leasing the car for two years with an option to buy at a fixed rate from ford motor credit . again , thank you for your assistance . respectfully yours , c . b . + + + _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ the auto program sells for $ 59 . 95 ,
but take advantage of this internet offer and order for only $ 29 . 95 ! ! ! + + + order the auto program now for $ 29 . 95 ( plus $ 2 . 90 s / h ) + + + order today and i will include two valuable reports , free ! 1 . getting good automotive service 2 . nine ways to lower your auto insurance c
ost order now : call to order toll free : 888 324-1873 or ( 614 ) 529-1390 pay with personal check , visa , or mastercard right over the phone . for any questions pertaining to the auto program , call ( 614 ) 529-1390 please be patient when calling to order . if you have any difficulty connecti
ng , please try again later . or send payment to : kevin cross 1773 hobbes drive hilliard , ohio 43026 i would like to say to anyone that might doubt whether the auto program will really work for them , i completely understand . i have bought a lot of how-to information manuals that were comple
te hype and did not have valuable information . if the auto program was not already proven to work , i would not have my name and a money-back guarantee in writing in the auto program . # # # # # # # # # # under bill s . 1618 title iii passed by the 105th u . s . congress , this e-mail is in co
mpliance with the law . to remove your name from our list , please reply to this e-mail with the subject line " remove . " we apologize for the inconvenience .\r\n'
```

Figure 11: Email text before cleaning

```
data_cut.iloc[2].text
```

```
'spam received mail expressed desire purchasing automobile recently visited one affiliate web site apologize mail unsolicited please scroll bottom message instruction declining targeted mail hello name kevin cross put auto program together show people method used obtain vehicle payment security
deposit needed bad credit want drive junker wanted drive nice late model vehicle always liked nice automobile long remember obtained last two automobile without put money le desirable credit automobile single biggest expense people make besides home unlike home automobile depreciate value putti
ng money downpayments security deposit car purchase lease smart although auto program initially put together people like bad credit program also valuable people excellent good credit since saving money would normally pay downpayment money security deposit read others say ordered auto program mo
nday friday obtained geo tracker money monthly payment money per month columbus ohio using method auto program obtained bmw convertible money thanks showing obtain great car even though bad credit columbus ohio note testimonial section full unsolicited letter customer used auto program whether
bad credit get financed excellent credit want eliminate downpayment security deposit without increasing monthly payment auto program work requirement must currently employed source income able make monthly payment time car salesman deal deal normal stress related dealing car salesman also mean
deal getting rejected bad credit take car older car really want easy use follow simple step step direction auto program obtain late model vehicle want easy auto program use explains everything need know obtain late model vehicle choice downpayment security deposit required pay normal monthly ca
r payment saving auto program save money money dollar since required put payment security deposit vehicle obtain risk money back guarantee money back guarantee writing auto program return auto program anytime within day full refund able obtain late model vehicle downpayment security deposit gua
rantee regardless credit history long use simple method auto program frequently asked question long take obtain vehicle receive auto program quickly actively use method provided auto program example eric brown columbus oh ordered auto program monday friday week called tell u obtained vehicle wa
nted program people bad credit auto program also people good credit want put normally required downpayment auto program show obtain vehicle choice money use money saved paying downpayment buy better thing lease car truck car dealership without payment pay monthly lease payment need auto program
many customer thought tried lease car dealership first must excellent credit lease car truck lease put security deposit first last lease payment capitalized cost reduction thing payment different name lease car dealership put buy car monthly car payment must high since downpayment required righ
t monthly car payment regular amount someone paid downpayment revolutionary method downpayment needed auto program good true step step method auto program approved lawyer market tested many satisfied customer confident benefit auto program put name money back guarantee behind testimonial mr kev
in cross great pleasure give warmest appreciation company providing necessary tool successfully secure late model chevy cavalier using st preferred method purchase program january secured vehicle january likewise one option available also opportunity assume lease late model acura legend assume
lease late model bmw course made wisest financially sound decision purchased chevy cavalier closing graciously thank wish continued success quest help others professionally dear gentleman writing tell good fortune acquiring new ford escort wagon ricart ford think program help instill confidence
could get new car least knew something piece junk driving thank assistance helping realize dream owning new vehicle still believe good fortune leasing car two year option buy fixed rate ford motor credit thank assistance respectfully auto program sell money take advantage internet offer order m
oney order auto program money plus money order today include two valuable report free getting good automotive service nine way lower auto insurance cost order call order toll free pay personal check visa mastercard right phone question pertaining auto program call please patient calling order d
ifficulty connecting please try later send payment kevin cross hobbes drive hilliard ohio would like say anyone might doubt whether auto program really work completely understand bought lot information manual complete hype valuable information auto program already proven work would name money b
ack guarantee writing auto program bill title iii passed th congress mail compliance law remove name list please reply mail line remove apologize inconvenience '
```

Figure 12: Email text after cleaning

References

1. C. (2019, July 11). Email Usage Statistics in 2019. Retrieved October 18, 2020, from https://www.campaignmonitor.com/blog/email-marketing/2019/07/email-usage-statistics-in-2019

2. Clement, J. (2020, June 24). Spam statistics: Spam e-mail traffic share 2019. Retrieved December 11, 2020, from https://www.statista.com/statistics/420391/spam-email-traffic-share

3. Clement, J. (2020, March 25). Number of e-mail users worldwide 2024. Retrieved December 11, 2020, from https://www.statista.com/statistics/255080/number-of-e-mail-users-worldwide/

4. Dada, E., Bassi, J., Chiroma, H., Abdulhamid, S., Adetunmbi, A., & Ajibuwa, O. (2019, June 10). Machine learning for email spam filtering: Review, approaches and open research problems. Retrieved October 12, 2020, from https://www.sciencedirect.com/science/article/pii/S2405844018353404

5. Das, M., Patel, H., & Samanta, S. (2020). Analysis of Machine Learning Algorithms for Email Classification Using NLP (No. 4107). EasyChair.

6. Drake, C. E., Oliver, J. J., & Koontz, E. J. (2004, August). Anatomy of a Phishing Email. In CEAS.

7. Forsey, C. (2020, July 22). The Ultimate List of Email Marketing Stats for 2020. Retrieved December 11, 2020, from https://blog.hubspot.com/marketing/email-marketing-stats

8. Graham, P. (2002, August). A Plan for Spam. Retrieved October 16, 2020, from http://www.paulgraham.com/spam.html

9. Kumar, A., Chatterjee, J. M., & Díaz, V. G. (2020). A novel hybrid approach of SVM combined with NLP and probabilistic neural network for email phishing. International Journal of Electrical and Computer Engineering, 10(1), 486.

10. Metsis, V., Androutsopoulos, I., & Paliouras, G. (2006, July). Spam filtering with naive bayes-which naive bayes?. In CEAS (Vol. 17, pp. 28-69).

11. Rennie, J. D. (2001). Improving multi-class text classification with naive Bayes.

12. Rennie, J. D., & Rifkin, R. (2001). Improving multiclass text classification with the support vector machine.

13. Sen, D., Das, C., & Chakraborty, S. (1970, January 01). [PDF] A New Machine Learning based Approach for Text Spam Filtering Technique: Semantic Scholar. Retrieved October 12, 2020, from https://www.semanticscholar.org/paper/A-New-Machine-Learning-based-Approach-for-Text-Spam-Sen-Das/11180267774e8d5168120a86dcfb1491556e2f24

Time Log

Date	Task	Time Spent (Minutes)	Time Spent (Hours)
10/7/2020	Topic Research	90	1.5
10/8/2020	Data Gathering	30	0.5
10/9/2020	Data Exploration and data Processing	120	2
10/12/2020	Literature Search	120	2
10/14/2020	Literature Search	45	0.75
10/15/2020	Literature Analysis	210	3.5
10/16/2020	Data Cleaning and Research	150	2.5
10/17/2020	Data Cleaning and Writing	180	3
10/17/2020	Proposal	180	3
10/18/2020	Proposal	540	9
12/1/2020	Data Addition and ETL for final dataset	120	2
12/2/2020	Data Normalization and Tokenization of data	120	2
12/3/2020	SVM and NB	120	2
12/5/2020	Neural Network	120	2
12/6/2020	Neural Network Fine tuning and data exporting	180	3
12/7/2020	Scripting and data evaluation an visualizations, outlining	480	8
12/8/2020	Rerun and fine tuning script, visualizations tweaking, final presentation work	480	8
12/11/2020	Final Report writing	360	6
12/11/2020	Final Report writing	240	4
	Total	3885	64.75

Code

ETL Export

```
import os
import pandas as pd
import mailbox
import bs4
path = '../Takeout/Mail/All mail Including Spam and Trash.mbox'
def remove_html(text):
    soup = BeautifulSoup(text, "html.parser", from_encoding="iso-8859-1")
    html_free=soup.get_text()
    return html_free

def get_html_text(html):
    try:
        return bs4.BeautifulSoup(html, 'lxml').body.get_text(' ', strip=True)
    except AttributeError: # message contents empty
        return None

class GmailMboxMessage():
    def __init__(self, email_data):
        if not isinstance(email_data, mailbox.mboxMessage):
            raise TypeError('Variable must be type mailbox.mboxMessage')
        self.email_data = email_data

    def parse_email(self):
        email_labels = self.email_data['X-Gmail-Labels']
        email_date = self.email_data['Date']
        email_from = self.email_data['From']
        email_to = self.email_data['To']
        email_subject = self.email_data['Subject']
        email_text = self.read_email_payload()
        temp = ''
        if type(email_text) == str:
            for i in email_text:
                temp+=i
            email_text = temp
        res =
{'labels':email_labels,'date':email_date,'email_from':email_from,'emial_to':email_to,'sub
ject':email_subject,'text':email_text}
        return res
    def read_email_payload(self):
        email_payload = self.email_data.get_payload()
        if self.email_data.is_multipart():
            email_messages = list(self._get_email_messages(email_payload))
        else:
            email_messages = [email_payload]
        return [self._read_email_text(msg) for msg in email_messages]
```

```
    def _get_email_messages(self, email_payload):
        for msg in email_payload:
            if isinstance(msg, (list,tuple)):
                for submsg in self._get_email_messages(msg):
                    yield submsg
            elif msg.is_multipart():
                for submsg in self._get_email_messages(msg.get_payload()):
                    yield submsg
            else:
                yield msg

    def _read_email_text(self, msg):
        content_type = 'NA' if isinstance(msg, str) else msg.get_content_type()
        encoding = 'NA' if isinstance(msg, str) else msg.get('Content-Transfer-Encoding',
'NA')
        if 'text/plain' in content_type and 'base64' not in encoding:
            msg_text = msg.get_payload()
        elif 'text/html' in content_type and 'base64' not in encoding:
            msg_text = get_html_text(msg.get_payload())
        elif content_type == 'NA':
            msg_text = get_html_text(msg)
        else:
            msg_text = None
        return (msg_text)
########################################################
import time
start_time = time.time()
mbox_obj = mailbox.mbox(path)
df = pd.DataFrame()
for idx,i in enumerate(mbox_obj):
    email = GmailMboxMessage(i)
    df = df.append(email.parse_email(),ignore_index=True)
    if idx % 100==0:
        print(idx)
df.to_csv('emails.csv')
print(time.time()-start_time)

df1 = pd.read_csv('emails.csv')
df1['class_labels']="
def joiner(temp_list):
    temp="
    for i in temp_list:
        temp+=i
    return temp
def stripper(row):
    mylist
=['Inbox','Unread','Category','IMAP_NotJunk','IMAP_$NotJunk','Opened','Important','IM
AP_$Junk','Category',"]
    try:
        i = re.split("\s|(?<!\d)[,.](?!\d)",row['labels'])
```

```
        for rec in mylist:
            if rec in i:
                i.remove(rec)
        if 'Spam' in i:
            return 'Spam'
        elif 'Travel' in i:
            return 'Travel'
        elif 'Purchases' in i:
            return 'Purchases'
        else:
            return joiner(i)
    except:
        pass
df1['class_labels']= df1.apply(stripper,axis=1)
#df1['class_labels'].value_counts()
#df1.drop(['target_label','new_labels'],axis=1,inplace=True)
class_list =['Promotions','Updates','Social','Personal','Purchases','Spam','Travel']
df_final = df1[df1['class_labels'].isin(class_list)]
df_final.to_csv('emails.csv')
```

Kaggle Datasets ETL

```
import pandas as pd
import os
myEmail = pd.read_csv('Data/emails_final.csv')
myEmail.drop(columns=['Unnamed: 0','labels'], inplace=True)
myEmail['ham_spam'] = ''
def ham_spam(row):
    if row['class_labels'] == 'Spam':
        row['ham_spam'] = 1
        return row['ham_spam']
    else:
        row['ham_spam'] = 0
        return row['ham_spam']
myEmail['ham_spam'] = myEmail.apply(ham_spam,axis=1)
class_list = ['Promotions','Spam']
myEmail = myEmail[myEmail['class_labels'].isin(class_list)]

extra = pd.read_csv('Data/messages.csv')
extra.label.value_counts()
extra_spam = extra[extra['label']==1]
extra_spam['txt_label']= 'spam'
spam_ham = pd.read_csv("Data/spam_ham_dataset.csv")
spam_ham.label_num.value_counts()
spam = spam_ham[spam_ham['label_num']==1]

data = {'subject':'','text':spam['text'],'class_labels':spam['label'],
'ham_spam':spam['label_num']}
```

```
append_1 = pd.DataFrame(data)
data = {'subject':
extra_spam['subject'],'text':extra_spam['message'],'class_labels':extra_spam['txt_label'
], 'ham_spam':extra_spam['label']}
append_2 = pd.DataFrame(data)
results = myEmail.append([append_1,append_2])

ham = results[results['ham_spam']==0]
ham['class_labels'] = 'ham'
spam = results[results['ham_spam']==1]
spam['class_labels'] = 'spam'
final = ham.append(spam)
final = final.sample(len(final)).reset_index(drop=True)
```

Final Dataset and Visualizations

```
# requirements
import string
import re
import pandas as pd
import textblob
import nltk.corpus
nltk.download('stopwords')
from nltk.corpus import stopwords
stop = stopwords.words('english')
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
nltk.download('punkt')
nltk.download('wordnet')
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
import sklearn.metrics
from sklearn.metrics import confusion_matrix
from sklearn.svm import SVC
import numpy as np
import keras
from keras.layers import Dense, Conv1D, Flatten, Dropout, Activation

#Functions
def remove_punc(row):
    punc = string.punctuation
    temp = ''
    for word in row['text']:
        if word not in punc:
            temp+=word
```

```
    row['text'] = temp
    return row['text']
def splitter_remover(row):
    temp = row['text']
    temp = temp.split()
    temp = [words for words in temp if len(words)>1]
    row['text'] = temp
    return row['text']
def stopWords(row):
    temp= [words for words in row['text'] if words not in stop]
    row['text'] = temp
    return row['text']
def word_stem(row):
    temp = [PorterStemmer().stem(i) for i in row['text']]
    row['text'] = temp
    return row['text']
from nltk.stem import WordNetLemmatizer
def word_lem(row):
    temp = [WordNetLemmatizer().lemmatize(i) for i in row['text']]
    row['text'] = temp
    return row['text']
def removeInt(row):
    temp = [word for word in row['text'] if not isinstance(word,int)]
    row['text'] = temp
    return row['text']
def strcon(row):
    temp = [str(item) for item in row['text']]
    row['text'] = temp
    return row['text']
def cleanup(row):
    temp=''
    for i in row['text']:
        temp+=i+" "
    row['text'] = temp
    return row['text']
def tester(y_pred, y_true):
    recall = sklearn.metrics.recall_score(y_true, y_pred)
    precision = sklearn.metrics.precision_score(y_true, y_pred)
    f1 = sklearn.metrics.f1_score(y_true, y_pred)
    fbeta = sklearn.metrics.fbeta_score(y_true,y_pred, beta=0.5)
    accuracy = sklearn.metrics.accuracy_score(y_true,y_pred)
    print('Accuracy', accuracy)
    print('Recall score', recall)
    print('Precision score', precision)
    print('f1 score', f1)
    print('fbeta score', fbeta)
    return [accuracy, recall, precision, f1, fbeta]

# import data
data = pd.read_csv('Data/final_data.csv', index_col=0)
```

```
ham = data[data['ham_spam']==0]
spam = data[data['ham_spam']==1]

test_size = [.25,.5,1,2,4]
data = {'model_name': [], 'accuracy': [], "recall": [], 'precision': []
          , "f1_score":[],'fbeta_score': [], 'ham_number': [], 'spam_number':[],
         'true_neg':[], 'false_pos':[], 'false_neg':[], 'true_pos': []}
df_final = pd.DataFrame(data=data)
for q in test_size:
   ham = ham.sample(n =round(len(spam)*q), random_state=1,
replace=True).reset_index(drop=True)
   data_cut = ham.append(spam)
   data_cut = data_cut.sample(len(data_cut),random_state=1).reset_index(drop=True)
   #data cleaning and feature engineering
   #lower text
   data_cut['text'] = data_cut['text'].str.lower()
   #extra whitespace
   data_cut['text'] = data_cut['text'].str.replace(r'\s+'," ")
   #next lines
   data_cut['text'] = data_cut['text'].str.replace(r'\\n'," ")
   data_cut['text'] = data_cut['text'].str.replace(r'\n'," ")
   data_cut['text'] = data_cut['text'].str.replace(r'\r'," ")
   #subject
   data_cut['text'] = data_cut['text'].str.replace('subject',"")
   #names
   data_cut['text'] = data_cut['text'].str.replace('jeremy randolph',"name")
   data_cut['text'] = data_cut['text'].str.replace('jeremy',"name")
   #phone number
   data_cut['text'] =
data_cut['text'].str.replace(r'^\(?[\d]{3}\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$',"contact number")
   #email addresses
   data_cut['text'] = data_cut['text'].str.replace(r'^.+@[^\.].*\.[a-z]{2,}$',"email")
   #currency
   data_cut['text'] = data_cut['text'].str.replace(r'£|\$',"money")
   #hyperlinks
   data_cut['text'] =
data_cut['text'].str.replace(r'\w+:\/{2}[\d\w-]+(\.[\d\w-]+)*(?:(?:\/[^\s/]*))*',"links")
   #removing numbers
   data_cut['text'] = data_cut['text'].str.replace(r'\d+(\.\d+)?'," ")
   #remove special chartacters
   data_cut['text'] = data_cut['text'].str.replace(r'[^a-zA-Z0-9]+'," ")
   #remove punctuation
   data_cut['text'] = data_cut.apply(remove_punc,axis=1)
   #split and remove single letter words
   data_cut['text'] = data_cut.apply(splitter_remover,axis=1)
   #stop words
   data_cut['text'] = data_cut.apply(stopWords,axis=1)
   #stemming to root words
   #data_cut['text'] = data_cut.apply(word_stem,axis=1)
   #lemmatization
```

```
    data_cut['text'] = data_cut.apply(word_lem,axis=1)
    #remove int
    data_cut['text'] = data_cut.apply(removeInt,axis=1)
    #make str
    data_cut['text'] = data_cut.apply(strcon,axis=1)
    #list to str
    data_cut['text'] = data_cut.apply(cleanup,axis=1)
    #train test split
    x_train,x_test,y_train,y_test = train_test_split(data_cut['text'],data_cut['ham_spam'],
random_state=1, test_size=0.1)
    #y_train = np.asarray(y_train)
    #y_test = np.asarray(y_test)
    #tfidf
    vector = TfidfVectorizer(sublinear_tf=True, max_df=0.5, stop_words='english')
    features_train = vector.fit_transform(x_train)
    features_test = vector.transform(x_test)
    #Naive Bayes
    nb = MultinomialNB()
    nb.fit(features_train, y_train)
    #score_train = nb.score(features_train, y_train)
    #score_test = nb.score(features_test, y_test)
    y_pred = nb.predict(features_test)
    #matrix = confusion_matrix(y_test,y_pred)
    #matrix
    '''import seaborn as sns
    from cf_matrix import make_confusion_matrix
    labels =['True Neg','False Pos','False Neg','True Pos']
    #ax =sns.heatmap(matrix, annot=True, fmt='d', group_names=labels)
    make_confusion_matrix(matrix, group_names=labels, figsize=(8,6))
    '''
    #svm
    kernal = ['linear','poly','rbf','sigmoid']
    for k in kernal:
        clf = SVC(kernel=k )
        clf.fit(features_train,y_train)
        y_pred =clf.predict(features_test)
        #print('Kernal: ',k)
        #tester(y_pred,y_test)
    #neural net
    model = keras.models.Sequential()
    model.add(Dense(64, activation='relu', input_shape=(features_test.shape[1],)))
    model.add(Dropout(0.5))
    model.add(Dense(32, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(16, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(8, activation='relu'))
    model.add(Dropout(0.2))
    model.add(Dense(4, activation='relu'))
    model.add(Dropout(0.2))
```

```
   model.add(Dense(1, activation='sigmoid'))
   #from keras.layers import Sequential
   model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
   #print(model.metrics_names)
   batch_size = 64
   epochs = 3
   history = model.fit(features_train, np.array(y_train), batch_size=batch_size,
epochs=epochs, verbose=1 )
   score = model.evaluate(features_test, np.array(y_test), batch_size=batch_size,
verbose=1)
   #print('Test loss:', score[0])
   #print('Test accuracy:', score[1])
   kernel = ['linear','poly','rbf','sigmoid']
   models = [nb, clf,model]
   models = {'Naive Bayes':nb, "Svm":clf, "Neural Network": model}
   data = {'model_name': [], 'accuracy': [], "recall": [], 'precision': []
       , "f1_score":[],'fbeta_score': [], 'ham_number': [], 'spam_number':[],
      'true_neg':[], 'false_pos':[], 'false_neg':[], 'true_pos': []}

   for names, model in models.items():
      if names != 'Svm':
         y_pred = model.predict(features_test)
         if names == 'Neural Network':
            y_final = []
            for i in y_pred:
               y_final.append(round(i[0]))
            res = tester(y_final,y_test)
            matrix = confusion_matrix(y_true=y_test,y_pred=y_final)
            data['true_neg'].append(matrix[0][0])
            data['false_pos'].append(matrix[0][1])
            data['false_neg'].append(matrix[1][0])
            data['true_pos'].append(matrix[1][1])
            data['ham_number'].append(len(ham))
            data['spam_number'].append(len(spam))
         else:
            res = tester(y_pred,y_test)
            matrix = confusion_matrix(y_true=y_test,y_pred=y_pred)
            data['true_neg'].append(matrix[0][0])
            data['false_pos'].append(matrix[0][1])
            data['false_neg'].append(matrix[1][0])
            data['true_pos'].append(matrix[1][1])
            data['ham_number'].append(len(ham))
            data['spam_number'].append(len(spam))
         count=0
         for key, point in data.items():
            if count <6:
               if key == 'model_name':
                  data[key].append(names)
               else:
                  try:
```

```
                    data[key].append(res[count])
                    count+=1
                except:
                    pass
        else:
            for i in kernel:
                clf = SVC(kernel=i)
                clf.fit(features_train,y_train)
                y_pred =clf.predict(features_test)
                res = tester(y_pred,y_test)
                matrix = confusion_matrix(y_true=y_test,y_pred=y_pred)
                data['true_neg'].append(matrix[0][0])
                data['false_pos'].append(matrix[0][1])
                data['false_neg'].append(matrix[1][0])
                data['true_pos'].append(matrix[1][1])
                data['ham_number'].append(len(ham))
                data['spam_number'].append(len(spam))
                count = 0
                for key, point in data.items():
                    if count <6:
                        if key == 'model_name':
                            data[key].append('Svm_'+i)
                        else:
                            try:
                                data[key].append(res[count])
                                count+=1
                            except:
                                pass

    df = pd.DataFrame(data=data)
    df_final = df_final.append(df)
    print(q)
df_final['total_average'] = (df_final.iloc[:,1:6].sum(axis=1))/5
df_final['ham_spam_ratio'] = round(df_final['ham_number']/df_final['spam_number'] ,2)
df_final.to_csv('final_stats.csv')

import matplotlib.pyplot as plt
import plotly.express as px
df_model = df_final
fig = px.bar(data_frame=df_model, x='ham_spam_ratio', y='total_average',
color='model_name',color_discrete_sequence=px.colors.qualitative.D3, title ='Overall
Model Performance by Ratio')
'''fig.update_traces(marker=dict(size=11, opacity=.8,line=dict(width=2,
                                        color='DarkSlateGrey')))'''
fig.update_layout(barmode='group')
fig.show()

df_model = df_final.groupby(by=['ham_spam_ratio'],
as_index=False)['total_average'].mean()
df_model.sort_values(by='ham_spam_ratio',ascending=True,inplace=True)
```

```
df_model['ham_spam_ratio'] =df_model['ham_spam_ratio'].astype(str)
#df_model
fig = px.bar( data_frame=df_model, x='total_average', y='ham_spam_ratio',
color='ham_spam_ratio',color_discrete_sequence=px.colors.qualitative.D3, title='Ham
Spam Ratio Total Average Performance')
#fig.update_traces(marker=dict(size=11, opacity=.8,line=dict(width=2,
                                #color='DarkSlateGrey')))
fig.show()

#confusion
import numpy as np
data = np.asarray([[410,0],[3,211]])
from cf_matrix import make_confusion_matrix
#make_confusion_matrix(data)
labels =['True Neg','False Pos','False Neg','True Pos']
#ax =sns.heatmap(matrix, annot=True, fmt='d', group_names=labels)
make_confusion_matrix(data, group_names=labels, figsize=(8,6),title='SVM RBF
Kernel: Ham Spam Ratio 2:1')

#best overall performance
df_model = df_final.groupby(by=['model_name'], as_index=False)
df_model = df_model.apply(lambda x: x.sort_values('total_average',ascending=False))
df_model = df_model.groupby('model_name').head(1)
df_model.reset_index(drop=True, inplace=True)
df_model.to_csv('best_model_performance.csv')

#best precison
df_model = df_final.groupby(by=['model_name'], as_index=False)
df_model = df_model.apply(lambda x: x.sort_values('precision',ascending=False))
df_model = df_model.groupby('model_name').head(1)
df_model.reset_index(drop=True, inplace=True)
df_model.to_csv('best_model_precision.csv')

#best fbeta
df_model = df_final.groupby(by=['model_name'], as_index=False)
df_model = df_model.apply(lambda x: x.sort_values('precision',ascending=False))
df_model = df_model.groupby('model_name').head(1)
df_model.reset_index(drop=True, inplace=True)
df_model.to_csv('best_model_fbeta.csv')

df_model = df_final
df_model['ham_spam_ratio'] =df_model['ham_spam_ratio'].astype(str)
fig = px.bar(df_model, x='model_name', y='fbeta_score',
facet_col='ham_spam_ratio',facet_col_wrap=10,title='Fbeta Analysis for Each Model')

fig.show()

df_model = df_final
df_model['ham_spam_ratio'] =df_model['ham_spam_ratio'].astype(str)
fig = px.bar(df_model, x='model_name', y='precision',
```

```
facet_col='ham_spam_ratio',facet_col_wrap=10,title='Precision Analysis for Each
Model')

fig.show()

df_model = df_final
df_model['ham_spam_ratio'] =df_model['ham_spam_ratio'].astype(str)
fig = px.bar(df_model, x='model_name', y='total_average',
facet_col='ham_spam_ratio',facet_col_wrap=10,title='Total Average Analysis for Each
Model')
fig.show()

df_model = df_final
df_model['ham_spam_ratio'] =df_model['ham_spam_ratio'].astype(str)
fig = px.bar(df_model, x='model_name', y='accuracy',
facet_col='ham_spam_ratio',facet_col_wrap=10,title='Accuracy Analysis for Each
Model')
fig.show()

df_model = df_final
df_model['ham_spam_ratio'] =df_model['ham_spam_ratio'].astype(str)
fig = px.bar(df_model, x='model_name', y='recall',
facet_col='ham_spam_ratio',facet_col_wrap=10,title='Recall Analysis for Each Model')
fig.show()

df_model = df_final
df_model['ham_spam_ratio'] =df_model['ham_spam_ratio'].astype(str)
fig = px.bar(df_model, x='model_name', y='f1_score',
facet_col='ham_spam_ratio',facet_col_wrap=10,title='F1 Analysis for Each Model')
fig.show()
```

Ham Spam Filtering

Classifying Emails by Text

By Jeremy Randolph

Introduction

- Email messaging makes up a large portion of personal communication .
- In 2020 alone there were 3.9 billion daily email users.
- In 2018 281.1 billion emails were sent and received on a daily basis.
- In March 2020 alone spam messages account for 53.95 % of all email traffic.
- Today, email spam filtering is performed in the background without a user noticing its work.
- Many older rule based filtering techniques have been successfully replace by machine learning algorithms.
- Modern systems claim to report a classification Rate of 99.99% with the implementation of machine learning.

Problem

- The objective of the project was to evaluate three popular machine learning algorithms for their effectiveness in email spam classification.
- Naive Bayes, Support Vector Machine, and a Neural Network was implemented for the task of classification .
- The implementation of these Machine learning algorithms aimed to attain similar results described in the literature review.

Literature Review

Machine learning for email spam filtering: Review, approaches, and open research problems.

- Evaluated the efficacy of several supervised learning techniques.
- Highlighted the importance of precision and the weight of false positives on model performance.
- Provided a rough framework as a starting point for project.

A novel hybrid approach of SVM combined with NLP and probabilistic neural network for email phishing.

- Provided a comparison of effectiveness between SVM and a Neural Neural network .
- Proposed a hybrid model utilizing both SVM and a Neural Network scoring better than if they were separate.

Literature Review (Cont.)

Analysis of Machine Learning Algorithms for Email Classification Using NLP

- Provided steps to consider when preprocessing email text using NLP
- Evaluated Several Supervised algorithms and provided metrics for each.

- Dada, E., Bassi, J., Chiroma, H., Abdulhamid, S., Adetunmbi, A., & Ajibuwa, O. (2019, June 10). Machine learning for email spam filtering: Review, approaches and open research problems. Retrieved October 12, 2020, from https://www.sciencedirect.com/science/article/pii/S2405844018353404
- Kumar, A., Chatterjee, J. M., & Díaz, V. G. (2020). A novel hybrid approach of SVM combined with NLP and probabilistic neural network for email phishing. International Journal of Electrical and Computer Engineering, 10(1), 486.
- Das, M., Patel, H., & Samanta, S. (2020). Analysis of Machine Learning Algorithms for Email Classification Using NLP (No. 4107). EasyChair.

Data File

- Personal Emails Processed from GMail provided 70k raw records to use.
- 40k where selected and processed for non spam (ham) instances.
- Spam instances bolstered from two Kaggle datasets to make up roughly 2k records for project.
- Final raw data file was saved as CSV for preprocessing.

subject	text	class_labe	ham_span
NEWSLETT	["See inside for NASM's February Newsletter.\n\r	ham	0
9 games p	['App Store\nYour next favorite game\nLooking fc	ham	0
=?utf-8?B	["Perfect 10 delivery\nEmail not displaying prope	ham	0
=?UTF-	["\n\n\nOffice Depot(R)Monumental Savings You	ham	0
	Subject: if you or someone you love suffers from	spam	1
	Subject: looking for a new date	spam	1
40% off at	['\n\nGap Outlet\nhttp://click.email.gapfactory.c	ham	0
$10 Bonus	["Click here to view your DICK'S Sporting Goods er	ham	0
Your Pers	["\nYour Personal YouTube Digest - Jun 28, 2012\n	ham	0

Approach : Data Cleaning

- Pandas dataframe used to store all data when being worked with.
- String functions applied to email data to remove all symbols, numbers, and other artifacts as wells substitution of any addresses, numbers or unique identifiers.

```
data_cut.iloc[2].text
```

```
'this is not spam . you have received this e-mail because you expressed a desire in purchasing an automobile , or you recently visited one of our affiliates web sites . we apologize if this e-mail is unsolicited . please scroll down to the bottom of this message for instructions on declining ta
rgeted e-mail . + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + hello , my name is kevin cross . i have put the the auto program together to show people methods that i have used to obtain vehicles with no down payment or security deposit needed . i had bad credit , but
i did not want to drive a junker ; i wanted to drive a nice late model vehicle . i have always liked nice automobiles as long as i can remember . i have obtained my last two automobiles without having to put any money down and with my less than desirable credit . an automobile is the single big
gest expense most people make besides there home . unlike homes , most automobiles depreciate in value . that is why putting out no money on downpayments or security deposits on a car purchase or lease is smart . although the auto program was initially put together for people like myself whom h
ave bad or no credit , this program is also valuable to people with excellent or good credit , since you will be saving the money you would normally have to pay in downpayment money or security deposits . _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ + + + read what others have to say : + + + "
i ordered the auto program monday and that friday i had obtained a 1995 geo tracker 4x4 with no money down and my monthly payments are only $ 179 per month " e . b . columbus , ohio . " using the methods in your auto program i obtained a 1997 bmw convertible with no money down , thanks for show
ing me how to obtain a great car even though i had bad credit . " m . h . columbus , ohio note : ( testimonial section has full unsolicited letters from customers that have used the auto program . ) whether you have bad credit and can not get financed , or if you have excellent credit and want
to eliminate the downpayment or security deposit without increasing the monthly payment , the auto program will work for you . requirements : you must be currently employed or have a source of income and be able to make monthly payments on time . that \'s it ! no car salesman to deal with ! you
do not have to deal with the normal stress related to dealing with a car salesman . this also means you do not have to deal with getting rejected because of bad credit or having to take a car that is older and not the car you really want . _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ easy to u
se just follow the simple , step by step , directions in the auto program and obtain the late model vehicle you want . that is how easy the auto program is to use . it explains everything you need to know to obtain the late model vehicle of your choice with no downpayment or security deposit re
quired . just pay normal monthly car payments . _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ savings to you : the auto program will save you between $ 500 to $ 5000 dollars since you will not be required to put a down payment or security deposit on the vehicle you obtain . _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ no risk , 100 % money-back guarantee the 100 % money-back guarantee is in writing in the auto program . you can return the auto program anytime within 30 days for a full refund if you are not able to obtain a late model vehicle with no downpayment or security deposit .
this guarantee is regardless of your credit history as long as you use the simple methods in the auto program . _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ frequently asked questions q : how long does it take to obtain a vehicle once i receive the auto program ? a : it is all up to you and ho
w quickly you actively use the methods provided in the auto program . example : eric brown from columbus , oh , ordered the auto program on a monday and on friday of the same week he called to tell us he obtained the vehicle he wanted . q : is this program just for people with bad or no credit
? a : no , the auto program is also for people with good credit who do not want to put down the normally required downpayment . the auto program shows you how to obtain the vehicle of your choice with no money down . so , you can use your money saved from not paying a downpayment to buy better
things ! q : i can lease a car or truck from a car dealership without a down payment . . . just pay monthly lease payments . why do i need the auto program ? a : many of our customers have thought that , until they tried to lease a car from a dealership . first , you must have excellent credit
to lease a car or truck . with a lease , you have to put down a security deposit , first and last lease payments , and capitalized cost reduction which is the same thing as a down payment . . . just a different name ! when you lease a car from a dealership , you have to put down the same or mor
e as when you buy a car ! q : the monthly car payments must be high since there is no downpayment required , right ? a : the monthly car payments are the regular amount . it is just the same as if someone paid a downpayment , but through our revolutionary method , no downpayment is needed . q :
is the auto program too good to be true ? a : the step-by - step methods in the auto program have been approved by my lawyer and have been market tested by many satisfied customers . i am so confident you will benefit from the auto program that i put my name and a 100 % money-back guarantee beh
ind it ! _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ testimonials mr . kevin cross , it is with great pleasure that i give my warmest appreciation to your company for providing me with the necessary tools to successfully secure a late model chevy cavalier using the 1st and most preferred meth
od . i purchase the program on january 3 , and secured a vehicle on january 6 ; likewise , i had more than one option available to me ! i also had to opportunity to assume the lease on a late model acura legend or assume the lease on a late model bmw . but , of course , i made the wisest and mo
st financially sound decision and purchased the chevy cavalier . . . . in closing , i graciously thank you and wish you continued success in your quest to help others . professionally , d . s . + + + dear gentlemen : . . . i am writing to tell you of my good fortune in acquiring a new ford esco
rt wagon from ricart ford . i think that your program did help instill some confidence in me that i could get a new car . at least i knew that i had to do something about the piece of junk i was driving . thank you for your assistance in helping me realize my dream of owning a new vehicle . i s
till can\'t believe my good fortune . i am leasing the car for two years with an option to buy at a fixed rate from ford motor credit . again , thank you for your assistance . respectfully yours , c . b . + + + _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ the auto program sells for $ 59 . 95 ,
but take advantage of this internet offer and order for only $ 29 . 95 ! ! ! + + + order the auto program now for $ 29 . 95 ( plus $ 2 . 90 s / h ) + + + order today and i will include two valuable reports , free ! 1 . getting good automotive service 2 . nine ways to lower your auto insurance c
ost order now : call to order toll free : 888 324-1873 or ( 614 ) 529-1390 pay with personal check , visa , or mastercard right over the phone . for any questions pertaining to the auto program , call ( 614 ) 529-1390 please be patient when calling to order . if you have any difficulty connecti
ng , please try again later . or send payment to : kevin cross 1773 hobbes drive hilliard , ohio 43026 i would like to say to anyone that might doubt whether the auto program will really work for them , i completely understand . i have bought a lot of how-to information manuals that were comple
te hype and did not have valuable information . if the auto program was not already proven to work , i would not have my name and a money-back guarantee in writing in the auto program . # # # # # # # # # # under bill s . 1618 title iii passed by the 105th u . s . congress , this e-mail is in co
mpliance with the law . to remove your name from our list , please reply to this e-mail with the subject line " remove . " we apologize for the inconvenience .\r\n'
```

Approach : Data Cleaning (Cont.)

- Natural Language Toolkit was used to further distill text data.
- All punctuations were removed.
- Stop words from email body were removed.
- Email bodies were lemmatized to reduce inflected words properly to retain greater meaning.

```
data_cut.iloc[2].text
```

```
'spam received mail expressed desire purchasing automobile recently visited one affiliate web site apologize mail unsolicited please scroll bottom message instruction declining targeted mail hello name kevin cross put auto program together show people method used obtain vehicle payment security
deposit needed bad credit want drive junker wanted drive nice late model vehicle always liked nice automobile long remember obtained last two automobile without put money le desirable credit automobile single biggest expense people make besides home unlike home automobile depreciate value putti
ng money downpayments security deposit car purchase lease smart although auto program initially put together people like bad credit program also valuable people excellent good credit since saving money would normally pay downpayment money security deposit read others say ordered auto program mo
nday friday obtained geo tracker money monthly payment money per month columbus ohio using method auto program obtained bmw convertible money thanks showing obtain great car even though bad credit columbus ohio note testimonial section full unsolicited letter customer used auto program whether
bad credit get financed excellent credit want eliminate downpayment security deposit without increasing monthly payment auto program work requirement must currently employed source income able make monthly payment time car salesman deal deal normal stress related dealing car salesman also mean
deal getting rejected bad credit take car older car really want easy use follow simple step step direction auto program obtain late model vehicle want easy auto program use explains everything need know obtain late model vehicle choice downpayment security deposit required pay normal monthly ca
r payment saving auto program save money money dollar since required put payment security deposit vehicle obtain risk money back guarantee money back guarantee writing auto program return auto program anytime within day full refund able obtain late model vehicle downpayment security deposit gua
rantee regardless credit history long use simple method auto program frequently asked question long take obtain vehicle receive auto program quickly actively use method provided auto program example eric brown columbus oh ordered auto program monday friday week called tell u obtained vehicle wa
nted program people bad credit auto program also people good credit want put normally required downpayment auto program show obtain vehicle choice money use money saved paying downpayment buy better thing lease car truck car dealership without payment pay monthly lease payment need auto program
many customer thought tried lease car dealership first must excellent credit lease car truck lease put security deposit first last lease payment capitalized cost reduction thing payment different name lease car dealership put buy car monthly car payment must high since downpayment required righ
t monthly car payment regular amount someone paid downpayment revolutionary method downpayment needed auto program good true step step method auto program approved lawyer market tested many satisfied customer confident benefit auto program put name money back guarantee behind testimonial mr kev
in cross great pleasure give warmest appreciation company providing necessary tool successfully secure late model chevy cavalier using st preferred method purchase program january secured vehicle january likewise one option available also opportunity assume lease late model acura legend assume
lease late model bmw course made wisest financially sound decision purchased chevy cavalier closing graciously thank wish continued success quest help others professionally dear gentleman writing tell good fortune acquiring new ford escort wagon ricart ford think program help instill confidence
could get new car least knew something piece junk driving thank assistance helping realize dream owning new vehicle still believe good fortune leasing car two year option buy fixed rate ford motor credit thank assistance respectfully auto program sell money take advantage internet offer order m
oney order auto program money plus money order today include two valuable report free getting good automotive service nine way lower auto insurance cost order call order toll free pay personal check visa mastercard right phone question pertaining auto program call please patient calling order d
ifficulty connecting please try later send payment kevin cross hobbes drive hilliard ohio would like say anyone might doubt whether auto program really work completely understand bought lot information manual complete hype valuable information auto program already proven work would name money b
ack guarantee writing auto program bill title iii passed th congress mail compliance law remove name list please reply mail line remove apologize inconvenience '
```

Approach : Feature Extraction

- The clean data set was split into testing and training data with a 90/10 split.
- NLTK provided a function to process text bodies into matrix representations using Term Frequency Inverse Term Frequency .
- TF-IDF goes further than just a count vectorization.
- Allows word relevancy to be represented in a document and provides a comparison to similar documents.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

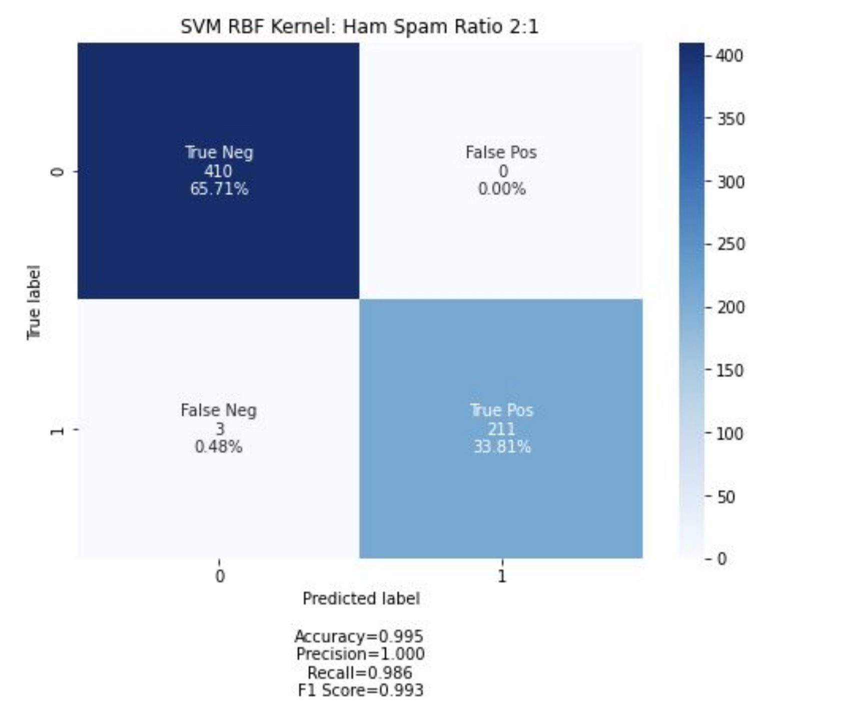
Approach: Classification and Final Data Collection

- The three algorithms were applied to the data in complexity order.
- Multinomial Naive Bayes and SVM where implemented From Sklearn Python packages.
- The neural network was developed using Keras packages that allows neural networks to be created sequentially.

- Once the three algorithms were created a script was developed to fine tune and develop a results dataset.
- Different ham to spam ratios were used to attain the best result possible.
- The ratios that was tested were 1:4, 1:2, 1:1, 2:1, and 4:1 for ham to spam instances.
- Metrics used to evaluate the test results where accuracy, recall, precision , F-1, and F-beta.

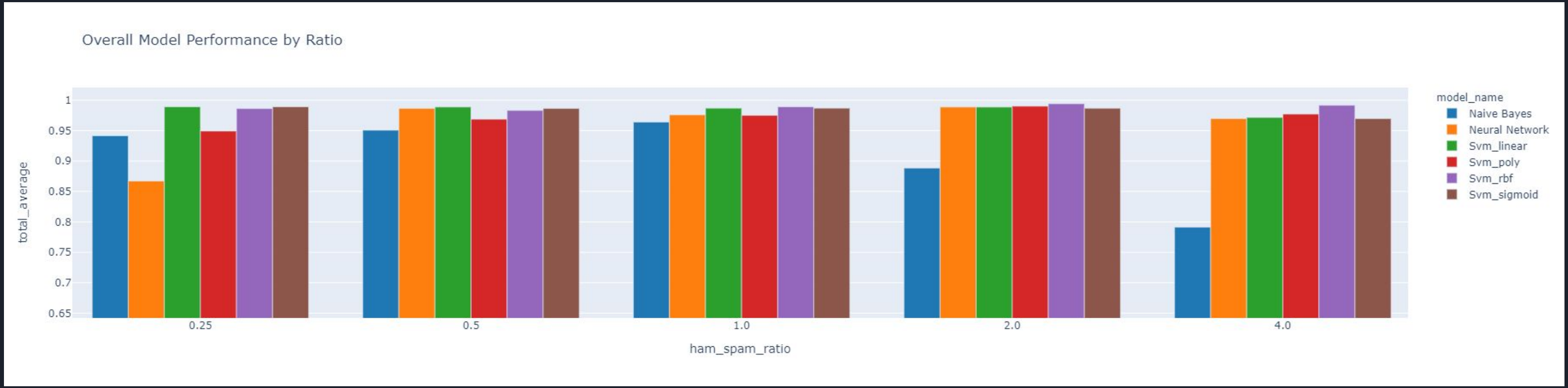
Results

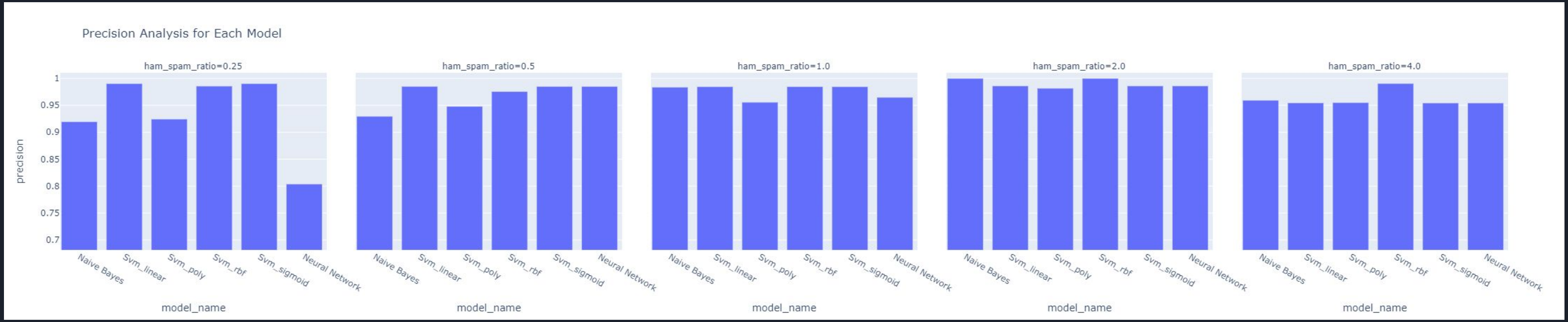
- SVM was the most effective in classifying email spam.
- In particular a perfect recall score was achieved using the RBF kernel and with a 2:1 Ham Spam Ratio.
- The neural network scored well but struggled to achieve the same results.
- Naive bayes worked well for being so simplistics however struggled as the number of instances increased.



Result (Cont.)

Overall Performance = The average of metrics tested

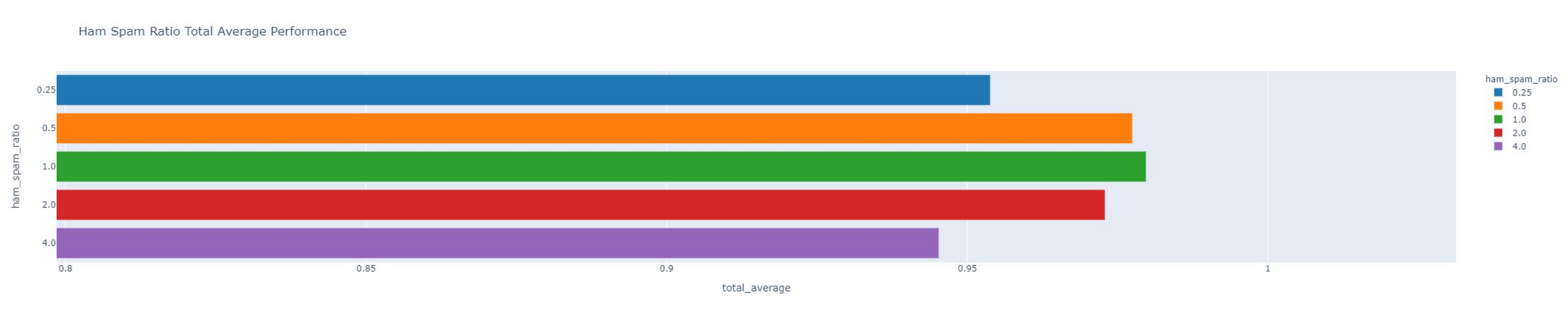




Discussion

- Each of the three classifiers performed well on the given task.
- The Neural network proved effective given time and effort to develop could be more effective than SVM at email classification.

- When increasing the ratio of ham to spam past 1:1 a drop off in overall performance was seen most notably the Naive bayes algorithm.
- SVM performed best within the ratios of 2:1 and 4:1. Further development could be spent on fine tuning this.



Conclusion

- This project aimed to apply three classification algorithms to the problem of spam classification.
- Naive Bayes struggled as the training data grew larger and the neural network performed better the more data trained on.
- SVM was able to perform the best on moderate amounts of data scoring high.
- Furthermore, the effort could be spent fine-tuning the ratio of ham mail to spam mail to get better results.
- Given more resources and time a better performing neural network could be created to outperform the SVM model used.

model_name	accuracy	recall	precision	f1_score	fbeta_score	ham_number	spam_number	true_neg	false_pos	false_neg	true_pos	total_average	ham_spam_ratio
Naive Bayes	96.394%	93.814%	98.378%	96.042%	97.430%	2079	2079	219	3	12	182	0.964119331	1
Neural Network	99.199%	99.065%	98.605%	98.834%	98.696%	4158	2079	407	3	2	212	0.988799501	2
Svm_linear	98.462%	99.043%	99.043%	99.043%	99.043%	520	2079	49	2	2	207	0.989267575	0.25
Svm_poly	99.359%	100.000%	98.165%	99.074%	98.527%	4158	2079	406	4	0	214	0.990249779	2
Svm_rbf	99.519%	98.598%	100.000%	99.294%	99.716%	4158	2079	410	0	3	211	0.994255851	2
Svm_sigmoid	98.462%	99.043%	99.043%	99.043%	99.043%	520	2079	49	2	2	207	0.989267575	0.25

-Highest scoring tests for each algorithm

Time Log

Date	Task	Time Spent (Minutes)	Time Spent (Hours)
10/7/2020	Topic Research	90	1.5
10/8/2020	Data Gathering	30	0.5
10/9/2020	Data Exploration and data Processing	120	2
10/12/2020	Literature Search	120	2
10/14/2020	Literature Search	45	0.75
10/15/2020	Literature Analysis	210	3.5
10/16/2020	Data Cleaning and Research	150	2.5
10/17/2020	Data Cleaning and Writing	180	3
10/17/2020	Proposal	180	3
10/18/2020	Proposal	540	9
12/1/2020	Data Addition and ETL for final dataset	120	2
12/2/2020	Data Normalization and Tokenization of data	120	2
12/3/2020	SVM and NB	120	2
12/5/2020	Neural Network	120	2
12/6/2020	Neural Network Fine tuning and data exporting	180	3
12/7/2020	Scripting and data evaluation an visualizations, outlining	480	8
12/7/2020	Rerun and fine tuning script, visualizations tweaking, final presentation work	480	8
	Total	3285	54.75

Questions?