**An Look At Linguistic Stylings Of Intrusive Messaging**

**Jacob Dineen, Mason David & Sam Harvey**

**School of Information Studies, Applied Data Science**

**Syracuse University**

|  |  |
| --- | --- |
| **Abstract**  This project will seek to analyze the linguistic stylings of spam messaging and statistically represent divergence from valid exchanges. Exploratory techniques, as well as supervised and unsupervised learning will be deployed to hypothesize about delineations between the two target classes. All data & scripts saved [here](https://drive.google.com/drive/folders/1QOcCpn3tJ4upafpE9r-4f0IXGblZMv2m).  **1. Introduction**  ***1.1. Overview***  The classic spam detection problem is often used as an introduction into the inner workings of nonparametric/parametric function approximation and machine learning - Both can be used synonymously to explain the construction of a paradigm that allows for a computer to learn without being explicitly programmed [3]. At its core, algorithms are tasked with finding some measure that allows for the mapping of a set of explanatory variables to a binarized representation of the target. With decision trees, we are looking to optimize a hierarchy dependent on feature importance, i.e. relative entropy/information gain, while support vector machines rely on maximizing a distance between a hyperplane (decision boundary) and the closest observations. Gradient based learning, in contrary, iteratively optimizes an objective function via backpropagation to find a set of weights that reduces overall error/loss. Where deep learning falters, in its current state, is a lack of interpretability brought along by the very thing that renders neural networks so successful in areas of computer vision and natural language processing. The interconnectedness of hidden layers and activations results in an entanglement of encoded representations with little explainability on what’s driving predictive capabilities (CNNs with attention & semantic mapping are current topics to come out of this field of research) [4].  Text mining is a process of transforming and analyzing, using descriptive and predictive solutions, an encoded representation of language.  The full dataset contains ~500,000 email exchanges. For the ability to quickly prototype models, we decided to sample this without replacement. Our sample is representative of ~1% of all collected data, or ~5000 total observations. Of those ~5000 observations we have a 71%/29% distribution of NotSpam/Spam. Downsampling or upsampling would be a viable measure to ensure/enforce class balance, but for now we’ll set a majority vote benchmark of 71% to compare our models against. Most performance will be measured using ‘Accuracy’ as our prime metric, however spam filters are likely judged more on their ability to minimize false positives, e.g. classifying valid emails as spam. It would be much more frustrating for an end user to discover that their real emails are being thrown into their spam folders as opposed to having to occasionally train the model (active learning) by moving a spam email from their inbox into their spam folder. Moving forward, the definitions for classification metrics will be defined broadly as:  False Positive: An email is not spam, but is classified as spam.  False Negative: An email is spam, but is classified as not spam.  We will be wanting to minimize the second type of error, otherwise known as a Type 2 error. Type 2 errors are present when the null hypothesis is false (spam) but our model fails to recognize it as such, or put more simply, spam is present but we fail to detect it [7].  **2. Method**  **2.1: Exploratory Analysis**  Exploratory Analysis is the means of describing, via through tables or visualizations, distributions or causal/non causal dependencies in your data. For the application of text mining, we identified two main delineations between our two classes during this step:  Real emails had a greater count of unique tokens, on average (**Figure 1**).  The average token count in each class:   * + Not Spam : 524 unique tokens   + Spam: 457 unique tokens | This project will center around the aforementioned solutions in an attempt to explain some of the style differences between our two classes, spam and not spam. Descriptive statistics will take a frequentist  approach to analysis by looking at such things as word and part of speech distribution, average string length and ultimately looking at each class’ parts of speech and their divergence from a uniform distribution (KL Divergence). Predictive solutions will center around supervised and unsupervised learning to identify features that are aggressively responsible for class mapping, and any natural boundaries being drawn in a high dimensional space. Error analysis, in conjunction with feature ranking (odds ratios/svm coefficients) will be conducted to derive hypotheses on why our classifier was unable to correctly predict a target class.  ***1.2. Project scope***  The dataset in question [2] is a collection of emails taken from high ranking Enron officials during an investigation by the Federal Energy Regulatory Commission. This is one of the most viable sources of email data available on the internet, as most ‘toy’ data is not truly representative of real world exchanges - This makes sense, holistically, as people are unlikely to want to share the contents of their legitimate emails, particularly in a business setting where the likelihood that you are bounded by nondisclosure is evident. It should be noted that this dataset was procured in the early 2000’s and it is likely that linguistics stylings share a causal relationship with time, changing at a fluctuating rate over given intervals. This could particularly be said about spam, which is adaptive. Companies responsible for sending out mass-email blasts are likely to adjust their ‘copy’ based on performance in the real world - If a particular email was to be marked as spam at bulk, which is a viewable metric by the third party, then they are unlikely to continue to use that exact style/representation for long stretches of time. This ties a bit into game theory and Generative Adversarial Networks - As these models get better at detecting spam the competitor (spam deliverers) get better at masking spam.  Individual tokens in Real emails were, on average, longer than tokens in spam emails.  The average character count in each token:   * + Not Spam : 6.25 characters   + Spam: 4.63 characters   We hypothesize that the lack of a true target audience is the reason that spam emails are generally shorter - They have no reason to go in depth on what they are pitching, as the longer one reads the more likely they are to get turned off of a product. Uncited research (forums) also speculate that the incoherence present in certain spam emails may in fact be purposeful in the avoidance of spam filters. By stringing together a bunch of unintelligible words/phrases, it is possible that it could formulate a representation more aligned with a particular class [8]. This is known as ‘word salad’ [9]. This could potentially explain the shortness of individual tokens in the spam class, but would take some more research to adequately confirm.    **Figure 1**  Summary Statistics also help to show us that most values fall within the interquartile range, so we are dealing with a outliers in our long tails. Most emails contain between 50 and 300 unique tokens.  Spam emails have a higher distribution of a lower use of unique tokens. |
| **2.2. Kullback-Leibler Divergence**  KL divergence, in short, is a measure to show the difference between two probability distributions. Borrowed from the field of information theory, it is littered across the theoretical underpinnings of machine learning it its current state. If you’ve ever built an iterative model via some python API, it’s likely that you’ve dealt with some variation of KL divergence. In tree-based algorithms it’s synonymous with relative entropy and information gain, which is the measure that helps you to optimize the hierarchy of feature nodes all the way down throughout your tree. In most other algorithms, such as logistic regression or neural networks, KL Divergence is engrained in cross entropy - Cross Entropy is how we measure our training loss during model construction. It is also how we iteratively optimize our weights via some form of gradient descent (We find the gradient of our loss times our learning rate and subtract it from our weights as we hillclimb to find a local minima). When you minimize your cross entropy loss, you are also minimizing your KL divergence between a true distribution (a one hot encoded vector of your target) and your predicted distribution (sigmoid or softmax output of your model). Simply put, it’s a very important metric [10].  To begin, we theorized that there may be some prevalent divergence between parts of speech within each of our classes that could act as a signal of whether or not something was spam. Preprocessing involved NLTK’s part of speech tagger and some liberal binning of those parts of speech into more comprehensible clusters (e.g. singular and plural nouns grouped into one bin called nouns). The figures below show the heads of our initial findings, transformed from raw counts to tuple probability distributions.  Spam POS Distribution    Not Spam POS Distribution      The two charts above are showing the same results in a different format. Some notable findings:   * Spam emails diverge from a uniform dist. regarding nouns/adjectives to a greater extent than not spam emails. * Real emails diverge from a uniform dist. regarding punctuation to a greater extent than spam emails.   This leads us to believe that spam emails might be comprised of a style suggestive of ‘selling’ based on the greater divergence when looking at adjective - i.e. Superfluous words that try to lure someone in. It also tells us that real emails likely place a higher emphasis on punctuation due to the perception of ‘professionalism’ inherently involved in grammatical correctness. It could also be reflective of the notion that workplace emails, from experience, try to get to the point in a more concise manner to deter from loss of attention. | On its own, this lends credence to the hypothesis that maybe these two types of emails have some different linguistic/grammatical styles that could lead them to be more easily identifiable. Still, as this is exploratory, it would prove valuable to show just how much these distributions diverge from each other.    *KL Divergence*  The formula above expresses KL Divergence in mathematical notation, where p(x) is our true distribution, and q(x) is some approximation/arbitrary distribution with which we can compare it to. For the sake of simplicity, q(x) was set to a uniform distribution, meaning that our probability distributions are equal to 1/n, 1-(1/n), where n is equal to the number of parts of speech tagged and binned at a previous step.  Suppose you have a true distribution you need to send somewhere, but the computational costs are high. KL Divergence provides a measure of what happens if you were to instead send over an arbitrary distribution If two distributions match exactly, our observed distribution and our arbitrary model (uniform/binomial, etc.) then our divergence is 0. It dives more into encoded representations and extra bits needed to encode a distribution then we’ll cover here.    After applying the stringtowordvector function the clustering can start. The results of clustering was not exactly what we were hoping for.    This graph shows that there is no real natural delimiter between our clusters. Ideally there would be two clean clusters but spam and not spam are not being classified correctly in our model. We had hypothesized that, assuming we were given clean clusters, we could use the attribute centroids to specify, in a high dimensional space, which words were causing the sub-groups of our target variable to be separated. Topic modelling via Latent Dirichlet Allocation may have been a viable alternative to better understand the word groupings which have a higher propensity to appear in one class vs the next. |
| **2.3. Text Clustering**  In our experiment we used filtered clustering in order to see if there was any real natural delimiters between the clusters. The first step in clustering was to merge our data set via python, creating a csv file that can be read into Weka. Once the data was read into weka there was some more refining that needed to be done to run the filtered clustering.  There are two different attributes in our data set, label and text. The label attribute was there to classify if an email was considered spam or not spam. The text attribute needed the stringtowordvector function to separate the text into individual attributes per word or number. After applying the stringtowordvector function the clustering can start.  features as independent while SVM will look at synergies or interactions between the features. In order to achieve this goal or objective, we also have to consider vectorization options. Vectorization allows us to understand our known words. We wanted to look at four different vectorizations which include Boolean, Term Frequency, NGram, and Term Frequency Inverse Document Frequency. Boolean is simply when our word counts are simply 1 or 0. NGram sets the range of n-values for different n-grams to be pulled. Term Frequency summarizes how often a term appears while TFIDF weights the terms based on how important that term is to the document with respect to all the documents. This helps weight important terms during the classification process as it tells which words are more important to determine the class. These will be the vectorizers used with our algorithms to determine if we can accurately predict the spam versus not spam class.  Next, we tested all of these in our python script. We pulled in the data from our files merged it into one big dataframe with our text and spam not spam class as a column. We did a 60 / 40 hold out split to train the data and test our model. After splitting the data, we defined the four vectorizers which include the list above. In these we also removed stop words and set a minimum frequency to 5.    Once these were set, we were able to train each model, score the model, print the confusion matrix, as well as error and feature ranked words. Below are the results from the different scores. We can see SVM’s constantly performed better than MNB. SVM with TFIDF performed the best with 99.2% accuracy. For this model, we included the confusion matrix as well. We can see how well it performed on this data set which was really well. | **2.4. Classification**  Text Classification is simply the process of trying to categorize text into different groupings. They’re multiple ways we can classify or categorize text data. We can can use simple classifications like in our case is Spam versus Not Spam. We could do document classification or even topic classification. Each of these represent categorizing our text into different groupings which allow us to use algorithms like Multinomial Naive Bayes or Support Vector Machines to possibly learn which set of text belongs to which class. In our project, we used these two algorithms to predict if a group of text was spam or not. In looking at the differences between these two algorithms, MNB is probabilistic while SVM is geometric. MNB also holds it’s  ***Boolean Vectorization Parameter***  SVM Score on Test data set 0.9705029013539652  MNB Score on Test data set 0.960348162475822  ***NGram12 Vectorization Parameter***  SVM Score on Test data set 0.9666344294003868  MNB Score on Test data set 0.9617988394584139  **Term Frequency**  SVM Score on Test data set 0.9680851063829787  MNB Score on Test data set 0.9637330754352031  ***TFIDF Vectorization Parameter***  SVM Score on Test data set 0.991779497098646  MNB Score on Test data set 0.968568665377176  *SVM Confusion Matrix*  [[1471 12]  [ 5 580]]  *SVM Precision Score / Recall Score*  [0.99661247 0.97972973]  [0.99190829 0.99145299]  Spam Not Marked errors: 5  Not Spam Marked Spam errors: 12  SVM Classification Report  precision recall f1-score support  0 1.00 0.99 0.99 1483  1 0.98 0.99 0.99 585  avg / total 0.99 0.99 0.99 2068 |
| After looking at this well of a model, we wanted to understand why if it could perform this well that why would it get any wrong? So we looked at the feature words per class and compared it to a couple misclassified text.    **Error: Spam Not Marked as Spam**  Subject: re time to reorder v ` icodin looking for vlcodln ? only place you can get it without prescription : glad 4 myrx now with same - day shipping , we are unbeatable . under - priced deals on other products like vallum too . deals wont last - visit us now !  Instantly, we noticed a mix of feature words in this misclassified result. We noticed words like Priced which was a spam word, but also words like deals which was a not spam word. In our opinion, this made it hard for the model to make the correct prediction.  Ultimately, this is a great model to have such a high accuracy. All models performed well, but SVM performed higher consistently. There is theory that this is commonly true especially with larger data sets as it captures the interactions between terms where as MNB captures them independently. This could very likely increase the models understanding of the text and in our experiment, it appears that way as it increased the prediction score. The weighting in TFIDF also helped increase the accuracy therefore causing us to conclude with a near perfect model.    **References**  Yingya paper for formatting.  Data:  [2]<http://www2.aueb.gr/users/ion/data/enron-spam/>  [3] <https://www.cs.cmu.edu/~./enron/>  [4] <https://distill.pub/2018/building-blocks/>  [5] <https://towardsdatascience.com/light-on-math-machine-learning-intuitive-guide-to-understanding-kl-divergence-2b382ca2b2a8>  [6] <https://www.nltk.org/book/ch05.html>  [7]Shermer, Michael (2002). [*The Skeptic Encyclopedia of Pseudoscience 2 volume set*](https://books.google.com/books?id=Gr4snwg7iaEC&pg=PA455&lpg=PA455&dq=type+ii+error+skepticism&source=bl&ots=bCBz8JJBTo&sig=FpB-13Igea9cS40ZZkP8CiAwxm8&hl=en&ei=3M4rTfq1EsWblgeK8oj_Cw&sa=X&oi=book_result&ct=result&resnum=5&ved=0CDQQ6AEwBA#v=onepage&q=type%20ii%20error%20skepticism&f=false). ABC-CLIO. p. 455. [ISBN](https://en.wikipedia.org/wiki/International_Standard_Book_Number) [1-57607-653-9](https://en.wikipedia.org/wiki/Special:BookSources/1-57607-653-9). Retrieved 10 January 2011.  [8]<https://www.forbes.com/sites/firewall/2010/03/17/the-most-common-words-in-spam-email/#1b21dbb67d80>.  [9]["Definition of "word salad"](http://oxforddictionaries.com/definition/english/word-salad?q=Word+salad). Oxford University Press. 2012  [10]<https://tdhopper.com/blog/cross-entropy-and-kl-divergence/> | **3. Conclusion & Further Work**  The classic spam detection problem is something people deal with everyday as emails that are not actually spam end up in spam folders when they shouldn’t.Our models are trying to correctly classify a sample data set of about 5,000 spam and not spam emails from senior management at Enron. One hypothesis was that spam emails were generally shorter in length and not nearly as in depth as not spam emails. We ran a multitude of different methods, including KL Divergence, POS tagging, clustering, and classification models on our data set. We got some positive results that we were very happy with and others that didn’t give us what we were looking for.  By running a KL Divergence model and Part of Speech Tagging there were two key takeaways from our data. First, we concluded that spam emails diverge from a uniform distribution regarding nouns and adjectives to a greater extent than not spam emails. Then the second takeaway was that real emails diverge from a uniform distribution regarding punctuation to a greater extent than spam emails. The filtered clustering was run to see if there was any real natural delimiters in the model. Unfortunately the clusters had both spam and not spam clustered together and did not give us the insight we were hoping for.Classification gave us a multitude of successful models as we ran multinomial naive bayes and support vector machines.All of our predictions gave us great results as all the scores on our data came in at over 95%. SVM performed the most consistently though throughout our experiments which goes with a theory that SVM captures the interactions between terms and MNB models captures the interactions independently. Ultimately, classification gave us a near perfect prediction model but did have a few keywords that made some emails get classified incorrectly. It wasn’t completely perfect and there are some small classification errors in our model but overall the results show that we can correctly predict between spam and not spam emails. |
|  |  |
|  |  |