Machine Learning in Email Spam Filtering

Problem Statement and Motivation

Email spam is an ever evolving entity that requires constant development in more efficient or completely new detection techniques (Ferrara, 2019). Spam is increasing in both quantity and sophistication with the help of AI (Ferrara, 2019). Machine Learning has been and can continue to be used to defend against spam. Due to the growing intensity of spam Machine Learning algorithms can be used to improve filtering. This paper goes over previous work with implementing Machine Learning algorithms in spam detection as well as my own work in utilizing and comparing a Naïve Bayes and Support Vector Machine approach. The data used in this approach is the publicly available Enron email corpus and all analysis is done in Python using publicly available packages.

Literature Review

Client-side spam filtering is a common technique that detects spam through pre-established lists. Yüksel, Çankaya, and Üncü look to Machine Learning to detect spam at the cloud level in an attempt to improve upon typical client-side filtering (Yüksel et al., 2017). The team uses Microsoft Azure and a Support Vector Machine (SVM) method to detect spam. Sending patterns of mail servers were analyzed and averaging sending time between "spam" and "good" emails were used to train the SVM model and classify further emails (Yüksel et al., 2017). The team found that using SVM they had a spam detection accuracy of 82.6% with a false positive rate of 17.3% (Yüksel et al., 2017). There are positives and negatives for this technique. I found the use of server sending patterns to be an interesting approach from the start. The use of SVM is a positive aspect as it can allow for soft margins. This could potentially be a benefit as the line between what is spam and what is a "good" is not always clear. I believe a con in this approach is identifying spam based on servers as some servers can send both spam and "good" emails.

Jiang looks to address the issue of the same servers sending both spam and "good" emails by using a model called Radial Basis Function network (RBF) (Jiang, 2007). Jiang used a frequency distribution for words in "good" emails and words found in spam (Jiang, 2007). RBF is used to define clusters and spam/good email classification is done based on those clusters (Jiang, 2007). This method uses more of a wholistic approach to analyzing email content. Every word (excluding punctuation) is factored into the classification which is a major pro. RBF is able to have a low misclassification rate and a high accuracy (Jiang, 2007). A con of this approach is that RBF is a relatively newer technique and has not had as much time to be proven. However, the results are quite good and for spam which is low consequence it is not a big risk. The use of new techniques is important to stay ahead of the ever-evolving spam methods.

Venkatraman, Surendiran, and Kumar explore a Naïve Bayes Classification technique. The team took a semantic similarity approach to spam detection which looks at lexical similarity coupled with a few different semantic techniques (Venkatraman et al., 2020). The Naïve Bayes looks at the probability of the words in a given email and determines the classification based on that probability (Venkatraman et al., 2020). The additional semantic methods break down this approach further with more advanced word relationship detection methods. The results the team found are rather impressive. The most advanced semantic approach coupled with Naïve Bayes returned at least a 96.5% accuracy (increasing

with spam word dataset size) (Venkatraman et al., 2020). The major con in this approach is the to generate a spam word dataset which can be time consuming and error prone. However, I feel the positives in this approach vastly outweigh the initial complex setup. With the highest accuracy of 98.89%, the level of precision using Naïve Bayes is well worth the cons.

Olatunji also looks at handling email spam using a SVM technique. Olatunji employs SVM for pattern recognition of email contents (Olatunji, 2019). The content patterns are what the algorithm uses to determine the classification of spam or not-spam. Email content analysis is similar to other methods previously discussed but is a dramatically different approach when compared to Jiang et al. where email server ending patterns were analyzed with SVM. Olatunji found an accuracy rate of about 95% with the SVM algorithm looking at content patterns (Olatunji, 2019). This is an improvement over the 85% Jiang et al. found using their own SVM technique. This improved accuracy is a considerable positive for this approach compared to the other SVM technique. The use of content pattern analysis makes sense intuitively. I could see a negative to this approaching being the use and collection of email contents required to develop this model. Email contents can be sensitive, personal, and private. The previous SVM method only looks at server sending and avoids the use of possible personal information.

Al-Tahrawi, Abualhaj, and Al-Khatib compare the performance of four different spam detection methods: Polynomial Neural Networks (PNN), SVM, Naïve Bayes (NB), and K-Nearest Neighbors (KNN). Each method was used to determine email content patterns based on language to classify email as either spam or not-spam (Al-Tahrawi et al., 2020). The team ran multiple tests with each method to test their performance in various scenarios. The main goal here is to compare the PNN performance against traditional state-of-the-art methods KNN, SVM, and NB (Al-Tahrawi et al., 2020). The overall findings showed that SVM performed the best when it comes to accuracy under all but the smallest of training sets (Al-Tahrawi et al., 2020). However, with exceptionally small training sets the PNN method was more accurate and was as accurate if not more accurate than KNN and NB (Al-Tahrawi et al., 2020). A positive to this approach is the high level of performance from small training sets. This is an advantage over other methods when data is in limited supply. A negative is that SVM scored marginally better in some metrics, making it a better choice when all else is held constant.

Summary of Work

The data originates from the Enron corpus containing over 600,000 individual emails. Ham and spam classified datasets are sourced from Natural Language Processing Group in the Department of Informatics at the Athens University of Economics and Business. The ham and spam datasets consist of about 16,000 emails each and were used to train the Naïve Bayes (NB) and Support Vector Machine (SVM) models.

Explanation of Model

Four models are used and compared in this work: Multinomial Naïve Bayes (mNB), Bernoulli Naïve Bays (bNB), Support Vector Machine (SVM), and Linear Support Vector Machine (lSVM). Each model approaches classification differently. In an attempt to determine which model works best the four models are compared and contrasted in their performance. To evaluate model performance the accuracy of all models is compared. The mNB and bNB models are both popular methods for semantic classification. The important distinction between the two models is that mNB uses term frequency, meaning how many times a term is used in the email increases its weight, while bNB uses term

occurrence, meaning the model is looking for how many unique terms show up in each email. Both SVM and lSVM also base their classification on terms but the key difference between these two models is that SVM utilizes a non-linear classification while lSVM uses a linear classification approach.

Explanation of Implementation and Experiment Setup

The ham and spam datasets are structured into two respective directories and each email is an individual text file. The text files are read into two respective ham and spam dictionaries using a loop that sets the key to the classification ham or spam, and the email content as a text string as the content. These dictionaries are further broken down into sub dictionaries where each word in the text string is "tokenized" where each word is broken out into its own element. These unique words then exist in a dictionary where they are classified as ham or spam.

The ham and spam dictionaries are combined into one and the result is shuffled to ensure an appropriate mix of ham and spam examples end up in the training and text datasets. The word/classification dictionary is used in training the models where the words are passed as the X variable and the classification of ham or spam is passed as the Y variable. mNB, bNB, SVC, and lSVM models are implemented from the Scikit-learn package along with other tools for performance analysis.

Results And Conclusion

In favorable training conditions (70% training 30% testing) Both Naïve Bayes models offered marginally greater accruacy (mNB 98.86%, bNB 96.47%) over the SVM Models (SVM 98.23%, lSVM 98.41%). However this picture changes as the training set is reduced. When the training set is reduced to 1% and testing set is increased to 99% the accuracy of all models decrease with bNB taking the largest hit (mNB 96.10%, bNB 84.79%, SVM 90.87%, lSVM 93.67%). Naïve Bayes still has the highest accruacy with mNB however the Support Vector Machine models do better than bNB.

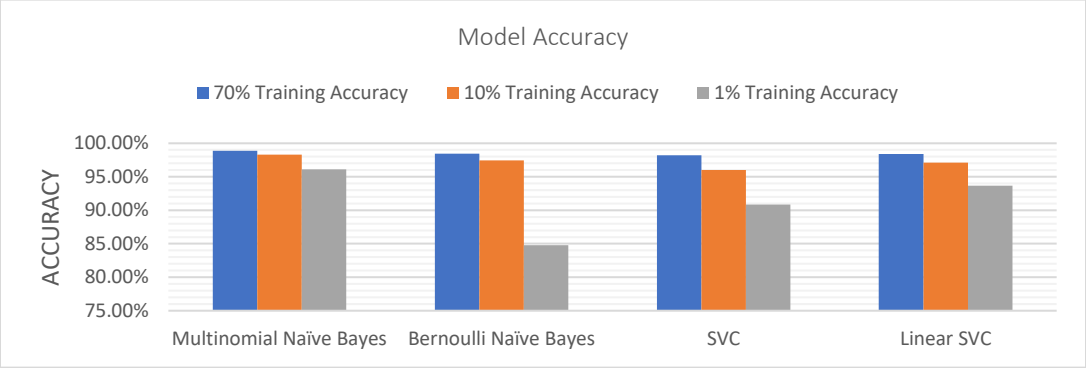

Figure A: Model Performance

Both Naïve Bayes and Support Vector Machine approaches offer relatively high accuracy in detecting spam emails. In the real world there is significantly more spam than there is training data, so a more model testing practical scenario is the low training and high testing metric. In this metric the Support Vector Machine algorithms did well as suggested by prior work. The Multinomial Naïve Bayes model did surprise well however accuracy is not the entire picture. Future work can be done to examine the precision, recall, and other scoring metrics of these models to paint a better picture. While accuracy is important, when it comes to email correspondence misclassifying email as spam (false positive classification) can have negative real-world implications.

Bibliography

Al-Tahrawi, M., Abualhaj, M., & Al-Khatib, S. (2020). Polynomial Neural Networks Versus Other Spam Email Filters: An Empirical Study. *TEM Journal*, *9*(1), 136–143. https://doi.org/10.18421/TEM91-19

Ferrara, E. (2019). The History of Digital Spam. *Communications of the ACM*, *62*(8), 82–91. https://doi.org/10.1145/3299768

Jiang, E. (2007). Detecting spam email by radial basis function networks. *International Journal of Knowledge Based Intelligent Engineering Systems*, *11*(6), 409–418.

Olatunji, S. O. (2019). Improved email spam detection model based on support vector machines. *Neural Computing & Applications*, *31*(3), 691–699. https://doi.org/10.1007/s00521-017-3100-y

Venkatraman, S., Surendiran, B., & Arun Raj Kumar, P. (2020). Spam e-mail classification for the Internet of Things environment using semantic similarity approach. *Journal of Supercomputing*, *76*(2), 756–776. https://doi.org/10.1007/s11227-019-02913-7

Yüksel, A. S., Çankaya, S. F., & Üncü, İ. S. (2017). Design of a Machine Learning Based Predictive Analytics System for Spam Problem. *Acta Physica Polonica, A.*, *132*(3), 500–504. https://doi.org/10.12693/APhysPolA.132.500