Naïve Bayes

CS460-G Machine Learning

Taylor Thomas

October 2021

Naïve Bayes

**Introduction**

As the world of communication via the internet and other digital mediums shows no signs of slowing, spam messages become more and more of a threat and continue to become more and more sophisticated. In order to prevent spam from taking time, energy, and possibly money away from victims of spam, implementing algorithms in email apps, text apps, and other areas where spam-detection can prove useful like online reviews, becomes increasingly important. In this project, the naïve bayes classifier is being used to calculate the probability of a new, unseen SMS/email example being spam and being not-spam, then comparing the two probabilities, and making a prediction based on the higher probability. Using naïve bayes means we’re assuming independence among the features, resulting in an algorithm where we can multiply these independent probabilities of features to obtain a probability of the entire example being either spam or non-spam. We are taking sorted data in from a text file and implementing the spam detection algorithm in python.

**Related Work**

* Almeida, Tiago A, Jurandy Almeida, and Akebo Yamakami. “Spam Filtering: How the Dimensionality Reduction Affects the Accuracy of Naive Bayes Classifiers.” Journal of internet services and applications 1.3 (2011): 183–200. Web.
  + High dimensionality of feature space proves to be a difficulty with the implementation of Naïve Bayesian approach to this problem of spam filtering. This means that reducing the dimensionality by eliminating and optimizing the features is an incredibly powerful way to make the Bayesian approach more efficient and practical. Dimensionality reduction can also reduce overfitting to training set because it optimizes the features of importance. This paper is useful for deciding how to handle for datasets with large enough email data that requires dimensionality optimization by focusing on looking at dimensions that will provide the most useful classification weight. The different classifiers are sensitive according to how they are dimensionally reduced.
* Guzella, Thiago S, and Walmir M Caminhas. “A Review of Machine Learning Approaches to Spam Filtering.” Expert systems with applications 36.7 (2009): 10206–10222. Web.
  + The evolution of the spam-detection problem is ongoing. Starting from hard-coded, non-automated updating algorithms pushed spammers to finding ways around these filters by slightly altering the characters of their spam messages to bypass the certain condition that usually flagged traditional spam. Machine learning offers the next era of spam-detection as it provides a way for constant updates to be made to the algorithm to still flag the new techniques used by spammers. This article specifically tackles the idea of image-processing algorithms vs traditionally text-processing algorithms, and after delving into the specific benefits of either or, comes to the conclusion that a merging of the two algorithms probably provides the best of both worlds in terms of optimizing for accuracy and taking into account processing time.

**Method**

The implementation of the training method in this program uses an input file with the following attributes:

* Each line represents a different example (SMS or email)
* The classification of each line is given by the word “ham” (non-spam) or “spam” located at the front of each line

This file is split 85:15 for training and testing data respectively. The first 85% of the data is split into two classes represented as lists of messages, where each message represents an SMS or email: spam and non-spam. Each of these classes is categorically analyzed according to the relative frequency of any given word that is found within a message in the class. This analysis is summarized in a dictionary where the keys correspond to the entire vocabulary of words in a class and the values are the relative frequencies (how many times each word appears across all messages in the class divided by the total number of words in the class).

With this dictionary, the program can now take in an unseen example and classify it according to naïve bayes algorithm.

Calculating the denominator is unnecessary as for each new example, they are equivalent and can be cancelled out as we are only interested in the relative probabilities of being spam and not spam. Calculating P(Spam) and P(Not Spam) is trivial as these values are just the frequency of any given example being spam or not spam based on the training set’s ratio of spam to non-spam messages.

Since independence is assumed for the features, P(X | Spam) and its not-spam counterpart is calculated by accessing each word in the new example message, X, and cross-referencing it with the spam and non-spam dictionary that we created, and accessing each relative frequency value associated with the key (word from X) and multiplying the series of values corresponding to each word in X (from the dictionaries) together to obtain a probability for any given string of words. In the case that a new word is introduced in an unseen example, we utilize a La Place smoothing idea where we essentially substitute the would-be 0% probability with 1 / 2\* total # of words in the given class, which fixes the problem where a 0 probability is possible, making the entire probability for the given example 0 just because it has one word which is unfamiliar to the training data.

After the first 85% of data in the file “spam-data.txt” is trained, the latter 15% is tested via the priorly explained method. Accuracy is printed, and the program asks if the user would like to input his/her own SMS/email input to be predicted based on the training done from the input file. The prediction is printed, and the prompt continues in a loop until the user answers “no” to the prompt.

**Experiments and Results**

*Spam/Ham Dataset:*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dataset** | **Train: Test Ratio** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| *Spam/Ham Dataset (given)* | 95:5 | 96.88% | 100% | 96.88% | .98 |
| *Spam/Ham Dataset (given)* | 90:10 | 97.22% | 100% | 97.22% | .99 |
| *Spam/Ham Dataset (given)* | 85:15 | 96.36% | 100% | 96.36% | .98 |
| *Spam/Ham Dataset (given)* | 80:20 | 95.86% | 95.86% | 95.86% | .98 |
| *Spam/Ham Dataset (given)* | 75:25 | 95.6% | 100% | 95.6% | .98 |
| *Spam/Ham Dataset (given)* | 50:50 | 94.81% | 100% | 94.81% | .97 |

**Review of Spam Detection Papers**

1. **“Spam Review Detection Using Deep Learning”**

* Shahariar, G. & Biswas, Swapnil & Omar, Faiza & Shah, Faisal & Hassan, Samiha. (2019). Spam Review Detection Using Deep Learning. 0027-0033. 10.1109/IEMCON.2019.8936148.

This paper delineates on an experiment with various techniques to correctly detect spam reviews on websites. At its core, the paper looks at the following traditional machine learning methods:

* Naïve Bayes
* K-Nearest Neighbors
* Support Vector Machine

and the following deep learning techniques:

* Multilayer Perceptron
* Convolutional Neural Network
* Recurrent Neural Network (Long Short-Term Memory) in this case

The paper explains the datasets that were used from Yelp and other hotel websites like Hotels.com, Trivago, TripAdvisor, some labeled, others not labelled as spam or non-spam. The data is first run through a pre-processing algorithm; labelled data is sent directly to phase 3 while active learning is implemented to create labels for the unlabeled examples.

Three different feature selections are used, including TF-IDF, N-grams, and Word embeddings. Then the datasets are sent to each of the listed algorithms for testing. It should be noted that training: testing ratio was also a variable that was tested.

Overall, the LSTM method provided highest overall accuracy achieving close to 97% with word embeddings technique. On the Yelp dataset, this method achieved an average of 95% with word2vec (word embeddings) technique across 60-90:40:10 train test ratios and 50:200 embedding dimensions.

Compared to the traditional classifiers, which averaged about 91.2% accuracy inn this Yelp dataset utilizing unigram and bigram techniques. Naïve bayes proved to be the most accurate traditional classifier, achieving slightly higher accuracy than SVM with constant variables.

This paper targeted a very practical use of these algorithms in the real-world. Outside of spam email and messages that can be filtered from people’s inboxes, introducing spam-filtering algorithms in the review sections of products on amazon and like-websites, hundreds of millions of fake reviews can be distinguished and banned from websites that falsely influence consumer opinion and either falsely slander competition or artificially inflate the perceived consumer opinion of one’s own product. This is generally an unregulated segment of online shopping, which does not often get as much attention as its potency asks for. Online shopping asks consumers to buy products with a minimal amount of tangible or useful information since consumers are generally used to making purchases after being able to see the item in real life. This fact means that consumer reviews are an incredible valuable source to consumers, and in turn, to companies that host these reviews. Product reviews are notoriously one of the most important aspects of the sale funnel when getting consumers to purchase a product online, since it provides (hopefully) genuine feedback from other users. This means these are incredible valuable data that can easily be artificially created by companies to increase their sales immorally.

Introducing these findings and mandating them in websites would be an incredible difficult task, with lots of backlash from online retailers, who have nothing to gain from this mandate, and without any ardent support from consumers, the implementation of these by the e-commerce stores is unlikely. Another solution would be developing a third-party software (possibly a browser extension) that takes reviews from a website, and classifies these reviews as either spam (fake) or not, and is targeted directly to the consumer to use. This may then influence e-commerce stores to implement their own algorithms in order to avoid coming across to the consumer as dishonest by artificially creating reviews.

1. **Machine learning for email spam filtering: review, approaches and open research problems**

-Emmanuel Gbenga Dada, Joseph Stephen Bassi, Haruna Chiroma, Shafi'i Muhammad Abdulhamid, Adebayo Olusola Adetunmbi, Opeyemi Emmanuel Ajibuwa,

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

Heliyon,

Volume 5, Issue 6,

2019,

e01802,

ISSN 2405-8440,

https://doi.org/10.1016/j.heliyon.2019.e01802.

(https://www.sciencedirect.com/science/article/pii/S2405844018353404)

This article analyzes various techniques like content-based filtering, which is most familiar, using Naïve Bayes, Support Vector Machine, K-Nearest Neighbors, etc. paired with various text-processing techniques. Heuristic/Rule based Spam filtering is defined as previously-created rules being the gate as to whether a new message is considered spam, where there is a defined threshold metricized by number of recognized patterns.

Like the previous paper led us to believe, this paper cites SVM as a more elegant and powerful solution to the classification problem. The reason for the success of this algorithm in regard to this problem is due to the fact that SVM, when implemented correctly, are able to identify patterns and classify them in a specific class very efficiently compared to other supervised learning methods.

Naïve Bayes is praised for its simplicity and its quickness compared to other, deeper-delving methods. It also requires fewer training data since it starts to approach a certain level of accuracy with the addition of more and more training examples.

One problem with the implementation of many of the algorithms is their susceptibility to security concerns. Keeping the data provided for the training of these algorithms secure is very important, as outsiders contaminating the data to provide a learning algorithm that is trained to let certain fake-examples pass at a high rate is possible.

This article attacks the various pros and cons of general techniques of spam-filtering. For example, the benefits of having feature free methods are apparent in the accuracies of such methods, but the computational costs weigh on the usability of these methods. Depending on where the methods are being implemented, the specifics of the algorithm become very important, and this article tries to connect the dots to some of the most common implementations and some of the scenarios they would be most effective in.

Focusing on the ideas and theories behind the mentioned techniques and methods for spam detection is important but pairing this paper with an experiment where the ideas and recommendations are hypothesized and run through the scientific method would give a lot more merit to the ideas mentioned in the article by providing specific figures form a controlled setting.

1. **Better Naïve Bayes classification for high-precision spam detection**

* Song, Yang, Aleksander Kołcz, and C. Lee Giles. “Better Naive Bayes Classification for High-Precision Spam Detection.” Software, practice & experience 39.11 (2009): 1003–1024. Web.

This paper hones in on the difficulties posed due to a particularly relatively high false-positive rate for spam filtering. It provides great reasoning for supporting more research in the area of Naïve Bayes since the algorithm requires a lot less resources in terms of computing and can scale fairly well due to the simplicity of it. So what ways can this be improved to get a better result without sacrificing the simplicity that causes NB to be so popular still?

One way to optimize is minimize the false positives because this would be classifying a real, possibly important email as spam, throwing the email in a spam folder where the user could possibly never see it.

Term weighting proves to perform well when implemented by itself, and also as an enhancer to other more common multiplicative functions. By focusing on how to improve short documents like emails and sms, the paper tackles an increasingly important area of spam-detection, since in general, most spam problems arise from shorter emails with less text and from text messages.