Spam Email Detection

Ussama Mustafa
Computer Science Department
College of Wooster
Wooster, OH
umustafa23@wooster.edu

Minh Nguyen
Computer Science Department
College of Wooster
Wooster, OH
mnguyen23@wooster.edu

Rekik Ziku
Computer Science Department
College of Wooster
Wooster, OH
rziku23@wooster.edu

Abstract

This research paper explores the performance of three distinct binary classifiers for spam detection: Logistic regression, Naive Bayes classifiers, and Support Vector Machines (SVMs). The study compares their computational complexity and practical applicability in pursuit of the most efficient and optimized classifier for tackling the spam detection challenge. The obtained results showed really high performance for all three models, with the naive Bayes classifier gaining a slight edge.

Introduction

Since its invention in 1972, email (Steinbrinck 2021) has become one of the most common and widely used communication tools worldwide. However, with the increasing number of emails being sent daily, the problem of spam emails has also become a major concern for individuals and organizations. Spam emails can be annoying, time-consuming, and sometimes even dangerous if they contain malware or phishing attacks.

This project develops a machine-learning model to effectively detect spam emails and filter them out of a user's inbox. This will be possible by using a dataset that classifies emails as spam or non-spam emails, pre-processing the data to extract relevant features, and training three models, each using a different classification technique: logistic regression (Chung 2020), naive Bayes (Raschka 2017), and support vector machines (SVMs) (Asimit et al. 2022). The resulting models will be used to accurately predict spam emails and their performance will be compared throughout the paper.

Dataset

For this project, we use a Kaggle dataset obtained through the Federal Energy Regulatory Commission as part of an investigation of Enron Corporation and its collapse. The dataset has two variables: "Category" and "Messages." There are 5572 messages that are categorized as either "spam" or "ham" (not spam.) Using this dataset, we train models to distinguish between spam and non-spam emails based on the features and patterns present in the Enron email data. The models we will be using are logistic regression, naive Bayes classification, and Support Vector Machine (SVM). We will compare these three models to determine which model performs best in identifying spam emails accurately. By comparing the results of these different models, we can identify the most effective algorithm for detecting spam emails and optimize its performance for future use.

Data Processing and Exploration

Since computers cannot understand raw text, we need to process the data and vectorize every word in the dataset. The data processing begins by loading the dataset into a dataframe using Pandas. To make things easier, the data is converted into lowercase. We then load a Python vectorizer or tokenizer which will convert the text data into numerical features that can be understood by the classification models that will be used. This conversion step was done by using a "CountVectorizer", an encoding algorithm used to convert text into a matrix of token counts, which represents the frequency of each word in the text.

Once the data is processed, it is good practice to perform data exploration to uncover aspects of the data that should be taken into account when using the trained models for predictions. An example of this would be a word cloud which shows the words that are most commonly used in spam and ham emails. As seen in Figure 1, words meant for advertising such as "free" or "now" are often found in spam mails, meanwhile words that are used in more normal conversations are more common in ham mails.

Logistic Regression

Logistic regression is a machine learning algorithm that is commonly used for binary classification tasks, where the model has to predict one of two possible outcomes. The basic idea behind logistic regression is to model the rela-

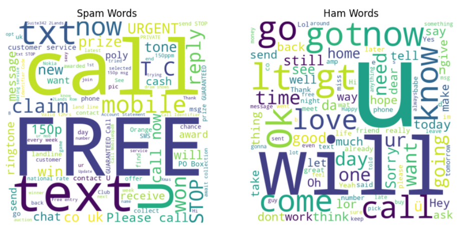

Figure 1: Word Clouds for Ham and Spam Emails

tionship between a set of input features and the probability of a particular outcome using a logistic function. The logistic function outputs a probability value between 0 and 1, which can then be used to make a prediction depending on a chosen threshold.

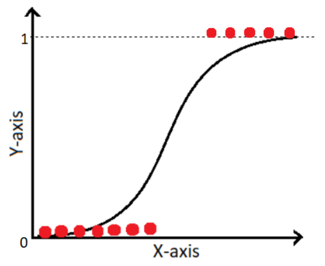

Figure 2: Logistic Regression Curve

As shown in Figure 2, a logistic regression curve can be used to model the relationship between a binary outcome variable and one or more predictor variables. For this project, the outcome variable is whether an email is classified as "spam" or "ham", and the predictor variables are features such as the presence of certain words or phrases or the length of the email. The logistic regression curve in this case would be used to estimate the probability that a given email is spam based on its predictor variables. The curve would have two possible outcomes: a high probability of spam or a low probability of spam, and the curve would be shaped like an S-curve, with the probability increasing as the predictor variables become more strongly associated with spam.

More mathematically, logistic regression can be written as:

$$P(y=1|x) = \frac{1}{1+e^{-xw}} \tag{1}$$

where y is the binary output variable (1 for spam, 0 for ham), $\mathbf{x}$ is the input feature vector, and $\mathbf{w}$ is the weight vector to be learned during training. The logistic function $\frac{1}{1+e^{-z}}$ maps any real number z to a probability between 0 and 1. The output of the logistic regression model is the predicted probability that the input $\mathbf{x}$ belongs to the positive class (i.e., spam).

During training, we provide the logistic regression algorithm with a labeled dataset of emails and their corresponding spam/ham labels. The algorithm learns to adjust the weights of the feature vectors to maximize the likelihood of the correct label for each email. Once we have trained the logistic regression model, we can use it to make predictions on new, unlabeled emails. We simply represent each new email as a feature vector and feed it into the model, which outputs a probability value between 0 and 1. If the probability value is above a certain threshold (e.g., 0.5), we classify the email as spam, otherwise, we classify it as ham.

Naive Bayes Classification

Naive Bayes classification is one of the most common algorithms used when approaching a binary classification task such as spam detection. The naive Bayes algorithm is built upon the assumption that every feature of the dataset is independent of the others. In the context of spam detection, to train a naive Bayes classifier, we calculate the prior probabilities of spam and ham emails in the training dataset and compute the conditional probabilities of words in spam or ham messages. At this point, we calculate the posterior probability of the message being ham or spam using Bayes' theorem and compare the resulting posterior probabilities to classify messages as spam or ham.

More mathematically, the classifier can be expressed as:

$$P(C_k|x_1, x_2, ..., x_n) = \frac{P(C_k)\prod_{i=1}^{n} P(x_i|C_k)}{\sum_{j=1}^{K} P(C_j)\prod_{i=1}^{n} P(x_i|C_j)} \tag{2}$$

where C_k is the k-th class, $x_1, x_2, ..., x_n$ are the n features of the input sample, and $P(C_k|x_1, x_2, ..., x_n)$ is the posterior probability of class C_k given the input sample x. The denominator is a normalization term that ensures the sum of all posterior probabilities equals 1. Given the assumption of conditional independence, the joint probability of the feature vector x given the class C_k can be written as the product of the conditional probabilities of each feature given the class:

$$P(x_1, x_2, ..., x_n|C_k) = \prod_{i=1}^{n} P(x_i|C_k) \tag{3}$$

The conditional probabilities $P(x_i|C_k)$ can be estimated from the training data using various techniques, such as maximum likelihood estimation or smoothing methods. In the context of spam detection, once the conditional probabilities are estimated, the classifier computes the posterior for the spam and ham classes and classifies the input email as the class with the highest probability.

Support Vector Machines

In the context of spam detection, Support Vector Machines (SVMs) are a type of classification algorithm that tries to find a line (or hyperplane) that separates the two types of emails in a way that maximizes the distance between the line and the closest examples. This distance is known as the margin, and SVMs are designed to find the hyperplane with the largest margin, as shown in Figure 3.

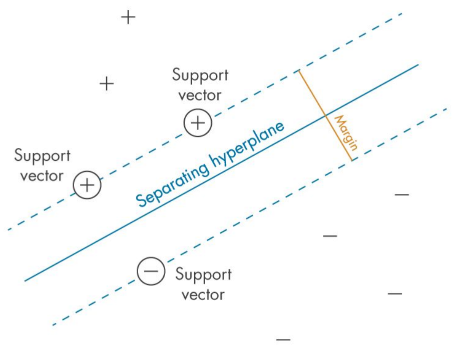

Figure 3: SVM Margin between 2 Classes

SVMs can be extended to use different types of functions to map the emails into a higher-dimensional space where they are more likely to be separable. These functions are known as kernels or filters, and they allow SVMs to capture more complex relationships between the emails and the labels.

Once the SVM has been trained on a dataset of labeled emails, it can be used to classify new emails as either spam or ham. The algorithm does this by computing a score for each email based on how closely it matches the hyperplane found during training. If the score is above a certain threshold, the email is classified as spam; if it's below the threshold, the email is classified as ham.

Method and Results

In order to build then evaluate the three classification models, the data was split into training and testing sets. The emails in the training set were then converted into a matrix of token counts, thus creating a feature vector for each email based on the frequency of each word in the message. The non-trained models of each classifier were loaded then trained on the training set and used to predict the categories of the messages in the testing set. The performance of each classifier was evaluated using the accuracy scores over the testing set as well as confusion matrices to show the true positives, true negatives, false positives, and false negatives of the classifier's predictions.

The obtained results show that all three classifiers perform well on the dataset, with an average accuracy over 0.97. However, the naive Bayes classifier provided the most accuracy score of 0.989. One possible explanation for the superior performance of naive Bayes is its assumption of feature independence, which is often violated in natural language processing tasks such as spam detection. However, this assumption may still hold in some cases, and the ability of naive Bayes to handle high-dimensional feature spaces may have contributed to its high accuracy in this dataset.

On the other hand, logistic regression and SVMs are more flexible in their ability to model complex relationships between features, which may explain their high accuracy scores in our study. Additionally, SVMs are known for their ability to handle non-linear decision boundaries, which could be important in distinguishing between spam and ham emails that are more difficult to separate.

Conclusion

The high accuracy scores obtained for all three methods in this study suggest that they are all viable options for spam detection. However, the choice of method may depend on the specific characteristics of the dataset and the desired trade-offs between computational efficiency and model complexity.

For instance, Naive Bayes tends to perform well on text classification tasks, especially when dealing with high-dimensional data, due to its simplicity and speed. Logistic Regression is also a popular choice for binary classification problems, but may be more computationally intensive than Naive Bayes. SVMs are a more complex model that can handle non-linearly separable data and have been shown to perform well in a variety of classification tasks, but may require more computational resources. Therefore, when choosing a method for spam detection, it is important to consider the specific needs of the application and to balance the trade-offs between model accuracy, computational efficiency, and model complexity.

Future work could explore hybrid approaches that combine the strengths of multiple methods for even higher accuracy in spam detection.

References

Asimit, V.; Kyriakou, I.; Santoni, S.; Scognamiglio, S.; and Zhu, R. 2022. Robust Classification via Support Vector Machines.

Chung, M. K. 2020. Introduction to logistic regression.

Raschka, S. 2017. Naive Bayes and Text Classification I - Introduction and Theory.

Steinbrinck, K. 2021. The History of Email: Major Milestones from 50 Years.