Real-Time Spam Email Detection

Abstract

The problem of detecting spam email is not new altogether. However, the email data has changed a lot over the period. Spam emails are evolving, and the system which detects them should also be. We propose a system to detect spam email in real-time. The time to train the model will be reduced as compared to offline classification. Typically, in offline classification, the training data resides in a text file. In online classification, the new data tuple is just added on top of the already existing data and a new model is generated for every instance. Count-min sketch is a data summarization technique which is nothing, but a probabilistic data structure that consolidates the data from an infinite stream of data. In order to compensate the obvious loss of data in Count-min sketch, we use Naïve-Bayes classifier to detect the spam email, which is strongly based on theory or basic principle. We thus evaluate the performance and accuracy of this method against the classical or offline approach of email classification.

Motivation

Training a model on a large set of data is quite challenging as we might not have the memory resource to hold all the data. On the other hand, we do get an infinite data from online streams. Emails are no exception to the above scenarios. It has large feature set, quite often it has large training data and it is a continuous streaming data. This motivates us to choose a sampling technique or data sketching which can most correctly represents the whole data. Stochastic sampling or even a regular time interval sampling will not necessarily work for real-world data like emails. There are some data summarization techniques like bloom filters. However, they do come with their own disadvantages. Count-min sketch is one such probabilistic data structure, which strives to represent the large pool of data by a constant size table. Since it is probabilistic, it is prone to error. However, to train a model in real-time, we do need to trade-off some level of correctness or accuracy for time.

Literature Survey

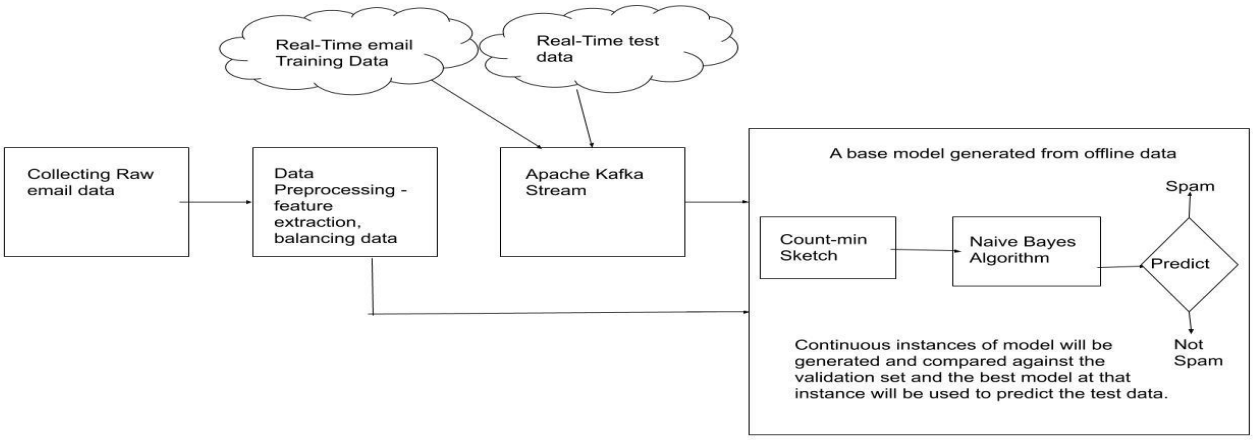
1) *A Sketch-Based Naive Bayes Algorithms for Evolving Data Streams,* 2018 IEEE International Conference on Big Data (Big Data): This paper has compared the evaluation metrics of different classification algorithms by applying count-min sketch or any other data summarization techniques. It also has given a great insight on approaching the problem of concept drift.

2) *An improved data stream summary: the count-min sketch and its applications.* This is the original paper which introduced the count-min sketch concept. It is one of the revolutionary ideas on data sketching techniques.

3) *Approximate Heavy Hitters and the Count-Min Sketch*. This is a lecture note from Professor Tim Roughgarden on implementing the Count-min sketch and insights on Heavy-Hitter problem.

Methodology

Experiment Design



Algorithms

1) Data processing Algorithms:

Bag of Words:

- It is an NLP technique to extract features from the text in e-mail and converting it into vectors for use in machine learning algorithms.

TF-IDF: (Term Frequency-Inverse Document Frequency)

- Term Frequency: To score the frequency of words in a current document.
- Inverse Document Frequency: To score how rare the word is across the documents. (TF-IDF Score = TF*IDF)
- TF-IDF score is used to create a training data which will be used to classify the emails.

2) *Naïve-Bayes Classifier:* It is based on the strong theoretical background. In data summarization, we are bound to lose some data which we of course try to compensate by choosing a classification algorithm which has a strong relationship to features.

3) *Count-min sketch:*

- A Data sketching technique.
- Count-min sketch uses constant size of memory to handle infinite stream of data. It has mathematical methods to identify the correct number of hash functions(w) and the number of buckets(d). The parameters ‘w’ and ‘d’ will determine the table size.

Evaluation Methods

We test metrics like accuracy, memory consumption and the time to train and test the model.

Accuracy

For accuracy we use F1-score method after building the Confusion Matrix. The reason for choosing this method is typically, positive cases will be less in "Spam email detection." So, we strive to increase the true positive cases while we also try to minimize the false positive cases. In other words, the classification algorithm will strive to prevent any non-spam email getting classified as spam. AUC-ROC method is not chosen because, it least cares about the positive and negative cases.

Memory

For simplification, we can choose a constant number of tuples from an infinite stream of data and to apply the Naïve-Bayes classifier, we can calculate the table size to store each attribute value in each tuple. Similarly, we can calculate the table size as we build the Count-min sketch table which is again a constant value. On comparing these two constants, we will get a fair idea on how memory efficient our algorithm is in building the training model.

Time

We compare the time taken in both training and prediction phase between the offline and online method. In this system, we can train the model as and when the new data arrives. But in case of offline method, the data typically resides in a text file and to train the new data, we are bound to repeat all the steps again. The proposed system can improve the accuracy of the model over a specific period with a minimal effort. Libraries like a profile in python also helps to analyze the CPU metrics to improve the algorithmic performance.

Deliverables

1) Code implementation and managing on GitHub.

2) Visualization tools using python libraries like matplotlib, etc.

3) To submit a research paper to a journal/conference. Our aim is to submit this research paper to IEEE computer.

4) Web application which receives data from Kafka/Spark. After receiving the data, the model is trained, and it will be associated with the corresponding data instance.

Milestones and Responsibilities:

ID	Milestone Description	Responsible Dept./Initials	Finished Week Plan	Forecast Week	+/-	Actual
1	Data Collection	Sivaranjani Kumar	Collecting email data	09/28	1-2 days	
2	Feature Selection and Extraction	Vignesh Kumar	Choosing the features from the email data collected	10/3	2-4 days	
3	Data cleaning and processing	Gulya	Removing noise and inconsistent data set	10/9	3-5 days	
4	Training Data	Pooja Patil	Obtaining the training dataset	10/12	2-4 days	
5	Implementing Count min Sketch	Akshaya Nagarajan	Implementing Count min Sketch using training data to create a frequency table	10/20	5-6 days	
6	Training the model using Naïve-Bayes	Sivaranjani Kumar	Naïve-Bayes classifier is used on the frequency table to classify the dataset	10/25	4-5 days	
7	Evaluation of accuracy	Vignesh Kumar	Assessing the model accuracy using evaluation methods like: Confusion matrix and F1-score	11/01	4-5 days	
8	Evaluation of memory and time	Pooja Patil	Memory efficiency is compared	11/08	3-4 days	

	consumption		between the count min sketch table size and actual data size.			
9	Fine Tuning the model	Gulya	Tune the model based on the evaluation results.	11/13	1-2 days	
10	Prediction	Akshaya Nagarajan	Test data will be passed on to the model to predict the result.	11/18	4-5 days	
11	Visualization of test data after classification	Pooja Patil Sivaranjani Kumar	Classified data will be plotted for visualization and accuracy.	11/23	5-6 days	
12	Web application integration	Teamwork	Integrating with a streaming framework like Kafka.	11/25	5-6 days	
13	Improvement/New scope	Teamwork	To Implement improvement ideas.	11/27	2-3 days	

Future Work/Improvement Ideas:

1) Planning to use Hashing Technique instead of count-min sketch.

2) Should use different classification algorithms.

3) Use ensemble learning to improve accuracy.

4) Apply concept-drift.