Chapter 3:

Conspiracy Detection Methodology

In this chapter we will describe the architecture of Conspiracy Detection Framework. There are mainly three sections in this chapter. First section 3.1 is about the system architecture of the Conspiracy Detection Framework where different modules of the architecture and relationship among them are described briefly. Among the main modules of the architecture like Mail Data Fetching module, storage module, Data Analyzing module, alert sending module. In section 3.2 we discuss about the analytical representation of our system which gives the details of the developed system with different algorithms, necessity flowcharts and tables required for analysis. Section 3.3 is about the complexity analysis of Conspiracy Detection Framework.

3.1 System Architecture

In this section of conspiracy detection model there are four module to work sequentially for analyzing the conspiracy. Data Acquisition and Refining module, Data Processing module, Training module, Testing in real time module. Here Data Acquisition and Refining module fetch the mail data from the database and then refine the data with unnecessary symbol, character and some other unwanted factors that will have no work with the detection model. Data Processing module reach the refined data to have the data with a matrix of frequency with respect to the importance in any pole. After finding the significance matrix the value of these significance goes to the classifier in this module of Training model. Now we have the trained module of conspiracy detection predictor. And the last and the most important module of this model is the testing with real time environment. In this module the data from the mail server is crawled by a crawler to give the input to the trained model. After analyzing this data model gives the answer or verdict to the management or monitoring body of any individual. The architecture is shown in below

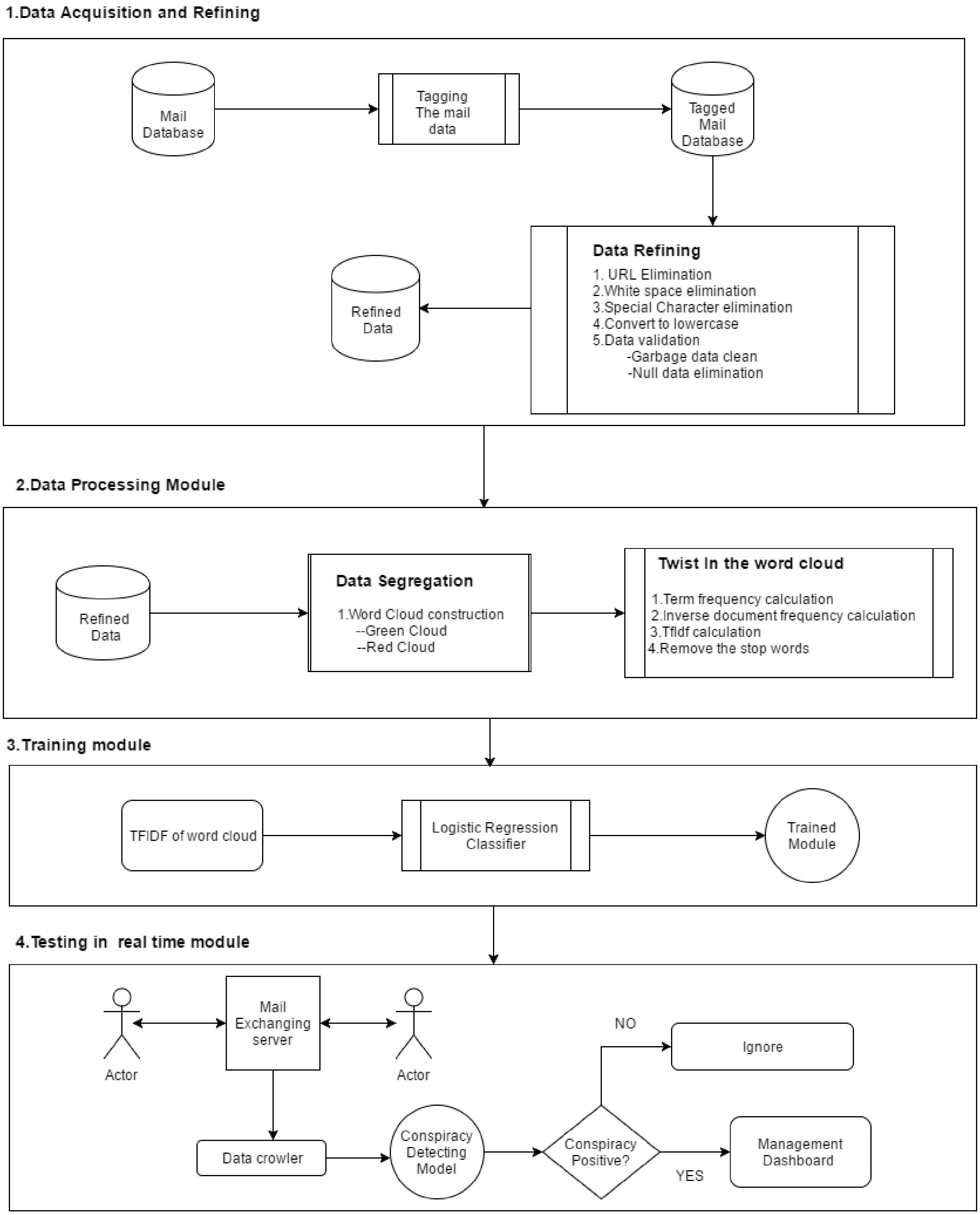


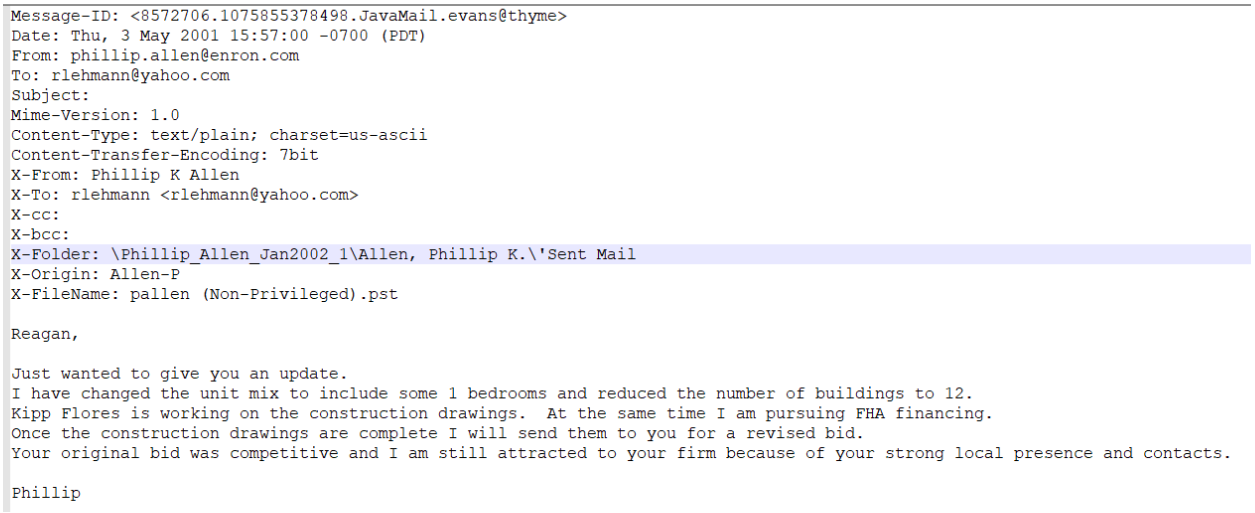
Fig 3.1: Conspiracy Predictor Model Architecture.

3.1.1 Data Acquisition and Refining

The very first module of this proposed model is *Data Acquisition and Refining*. This model is used to level the dataset with green and red level. This tagging makes the dataset leveled properly and segregate the body section. After tagging the dataset with conspiracy infected mail and conspiracy uninfected mail, the mail data is stored in a csv file. Now we have the dataset stored in the form of a comma-separated values file with mail data and corresponding category. Category is defined by 0 and 1. 0(zero) means the Green data and 1(one) means the Red data in the dataset.

After saving the leveled data in a .csv extension file the data is then import to the module to analyze and refine. The data is the refined with some necessary steps. The dataset is a mixture of words, emoticons, symbols, URLs and references to people.

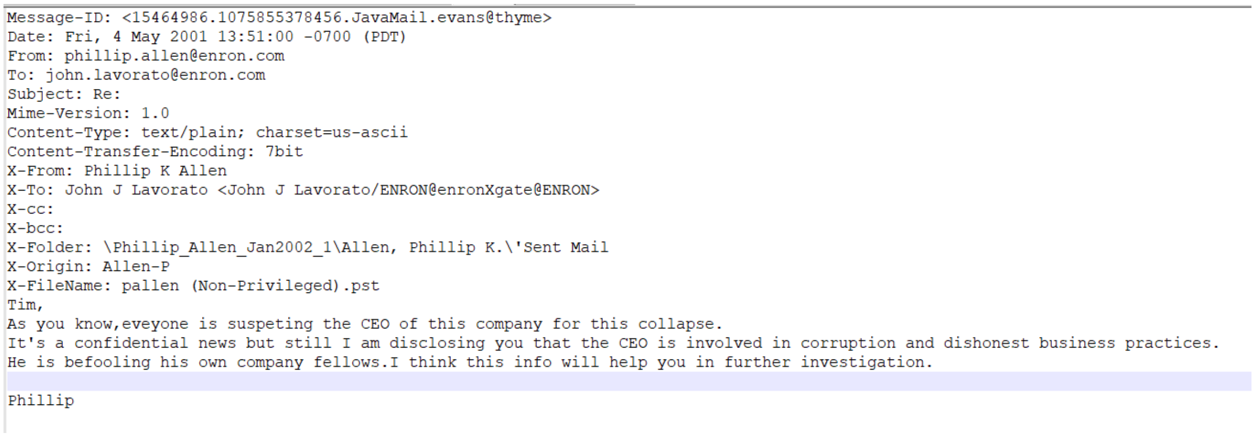
In the first category of Green set the mail body can reflect so many affair in the sentiment. Such as office affair, request mail, satisfaction mail, gratitude sentiment mail and some more documentation mail. Fig 2 shows the Green mail data in csv file



```
Message-ID: <8572706.1075855378498.JavaMail.evans@thyme>
Date: Thu, 3 May 2001 15:57:00 -0700 (PDT)
From: phillip.allen@enron.com
To: rlehmann@yahoo.com
Subject:
Mime-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
X-From: Phillip K Allen
X-To: rlehmann <rlehmann@yahoo.com>
X-cc:
X-bcc:
X-Folder: \Phillip_Allen_Jan2002_1\Allen, Phillip K.\'Sent Mail
X-Origin: Allen-P
X-FileName: pallen (Non-Privileged).pst

Reagan,

Just wanted to give you an update.
I have changed the unit mix to include some 1 bedrooms and reduced the number of buildings to 12.
Kipp Flores is working on the construction drawings.  At the same time I am pursuing FHA financing.
Once the construction drawings are complete I will send them to you for a revised bid.
Your original bid was competitive and I am still attracted to your firm because of your strong local presence and contacts.

Phillip
```

Fig 3.2: Green Mail Content

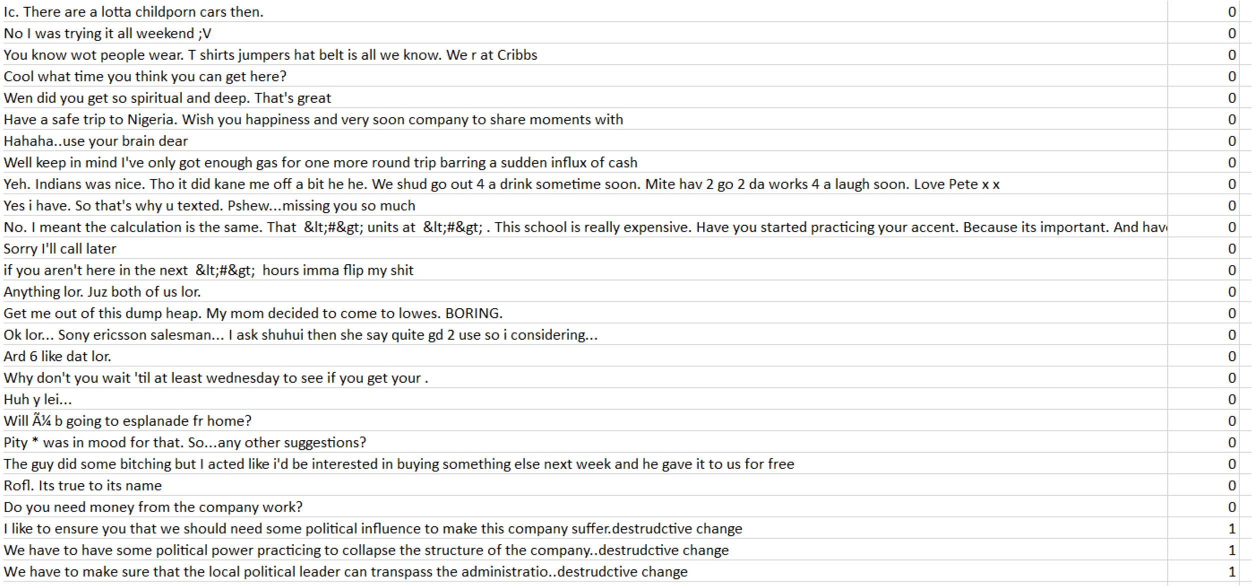
In the next category of the Red set of mail infected with office conspiracy. This category is leveled by one.This sort of mail data reflect rage, dissatisfaction, overpower intention etc that means the overall bad effect for the company. Fig 3 shows the Red mail category



```
Message-ID: <15464986.1075855378456.JavaMail.evans@thyme>
Date: Fri, 4 May 2001 13:51:00 -0700 (PDT)
From: phillip.allen@enron.com
To: john.lavorato@enron.com
Subject: Re:
Mime-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
X-From: Phillip K Allen
X-To: John J Lavorato <John J Lavorato/ENRON@enronXgate@ENRON>
X-cc:
X-bcc:
X-Folder: \Phillip_Allen_Jan2002_1\Allen, Phillip K.\'Sent Mail
X-Origin: Allen-P
X-FileName: pallen (Non-Privileged).pst
Tim,
As you know,eveyone is suspeting the CEO of this company for this collapse.
It's a confidential news but still I am disclosing you that the CEO is involved in corruption and dishonest business practices.
He is befooling his own company fellows.I think this info will help you in further investigation.

Phillip
```

Fig 3.3: Red Mail content

And the overall csv file will look like these in figure 4



Ic. There are a lotta childporn cars then.	0
No I was trying it all weekend ;V	0
You know wot people wear. T shirts jumpers hat belt is all we know. We r at Cribbs	0
Cool what time you think you can get here?	0
Wen did you get so spiritual and deep. That's great	0
Have a safe trip to Nigeria. Wish you happiness and very soon company to share moments with	0
Hahaha..use your brain dear	0
Well keep in mind I've only got enough gas for one more round trip barring a sudden influx of cash	0
Yeh. Indians was nice. Tho it did kane me off a bit he he. We shud go out 4 a drink sometime soon. Mite hav 2 go 2 da works 4 a laugh soon. Love Pete x x	0
Yes i have. So that's why u texted. Pshew...missing you so much	0
No. I meant the calculation is the same. That &lt;#&gt; units at &lt;#&gt; . This school is really expensive. Have you started practicing your accent. Because its important. And hav	0
Sorry I'll call later	0
if you aren't here in the next &lt;#&gt; hours imma flip my shit	0
Anything lor. Juz both of us lor.	0
Get me out of this dump heap. My mom decided to come to lowes. BORING.	0
Ok lor... Sony ericsson salesman... I ask shuhui then she say quite gd 2 use so i considering...	0
Ard 6 like dat lor.	0
Why don't you wait 'til at least wednesday to see if you get your .	0
Huh y lei...	0
Will Ã¼ b going to esplanade fr home?	0
Pity * was in mood for that. So...any other suggestions?	0
The guy did some bitching but I acted like i'd be interested in buying something else next week and he gave it to us for free	0
Rofl. Its true to its name	0
Do you need money from the company work?	0
I like to ensure you that we should need some political influence to make this company suffer.destrudctive change	1
We have to have some political power practicing to collapse the structure of the company..destrudctive change	1
We have to make sure that the local political leader can transpass the administratio..destrudctive change	1

Fig 3.4: A Peek into the Dataset

3.1.1.1 Data Refining

In our dataset we have the mail with some noise. Noises in the mail is natural because people send so many things in the mail to express his opinion. In mail data there are some link, URLs, emotion some unwanted symbol to refine .This raw data should be refined before training the model. To get the best out of our dataset, we applied a number of data cleaning process. At first some general cleaning are done, such as:

- Every sentence is first converted into lowercase format.
- Two or more spaces are replaces with a single space
- Quotes (" and ’), extra dots (.) and spaces are stripped from the ends of sentences.
- Null data elimination as well as the garbage data.

To handle the special component of a sentence, we have done the following pre-processing tasks:

1. **URL:** Users often sends URL in their mail. In our training, any particular URL doesn’t contain any special feature and if we kept the URLs in the sentences, that would have been leaded to sparse feature. Therefore, we remove all the URL from the sentences. To match the URLs we have used this regular expression ((www\.[\S]+)|(https?://[\S]+)).
2. **Special Cleaning:**
    - Any punctuation [’"?!,.():;] from the word is stripped. Words with three or more letter repetitions are converted to two letters.
    - Some people send their mail like I am happppppy which adds multiple characters on a certain words. Mail containing this type of words are handled by converting the word happppppy to happy.
    - To handle the words like sugar-free and our’s, we have removed - and ’. This type of words are converted into a more general form like sugarfree and ours.
    - Then we checked for valid word by checking successive alphabets, if it is not valid then we have stripped them.

3. **Contracted Word Handling:** Users often sends mails containing words in contracted form. Like are not is written as aren’t, I am is written as I’m etc. We converted the contracted word to their long form. A list of contracted word and their long form are given in Table 3.1:

Table 3.1: Contracted Word and Long Form

Contracted form	**Long form**	**Contracted form**	**Long form**
Aren’t	Are not	I’m	I am
Isn’t	Is not	Weren’t	Were not
Haven’t	Have not	Hasn’t	Has not
Hadn’t	Had not	Won’t	Will not
Don’t	Do not	Doesn’t	Dose not
I’d	I had	I’ll	I shall
They’ll	They will	He’ll	He will
She’ll	She will	Can’t	Can not
Mustn’t	Must not	Shouldn’t	Should not
Couldn’t	Could not	Wouldn’t	Would not

3.1.2 Data Processing Module

In this module the cleaned mail data is being further processed with some algorithmic process. Such as vectorization, featuring and stop word removing. By following these process the mail data will be ready for using in the classifier to train our predicting model. These following steps are described below.

3.1.2.1 Tokenization:

Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called tokens, perhaps at the same time throwing away certain characters, such as punctuation. Here is an example of tokenization:

Input: Mr .X can we meet today. Some documents should be reviewed

Output: Mr, X, can, we, meet, today, some, document, should, be, reviewed.

These tokens are often loosely referred to as terms or words, but it is sometimes important to make a type/token distinction. A token is an instance of a sequence of characters in some particular document that are grouped together as a useful semantic unit for processing. A type is the class of all tokens containing the same character sequence. A term is a (perhaps normalized) type that is included in the IR system's dictionary.iii

These tokens are further used to make feature vector.

3.1.2.2 Feature Vector:

We have produced our dataset based on the working place conspiracy theory from a pronounced journal. Here we have come to know about the details about the conspiracy and its related outcome, concept, outcome and immediate effect to detect from the work place. So a common style is found in the mail data during producing the mail data set.

A common style based analysis method is called writer invariant or author invariant. It claims that all texts written by the same author are similar or invariant. In other words texts written by the same author will be more similar than those written by different ones. Even though we are not interested in the authors of tweets, style based features is an important analysis method, since the topic all employee write about is similar and the purpose of the mail is the same, mainly to make successful their propaganda, it is reasonable to believe that the style of writing they have might be similar. We used frequency of words, presence of words, count of words and use of punctuations as style based features.

We have used TF*IDF to represent our features. TF*IDF is an information retrieval technique that weighs a term's frequency (TF) and its inverse document frequency (IDF). Each word or term has its respective TF and IDF score. The product of the TF and IDF scores of a term is called the TF*IDF weight of that term. It shows the relative importance of a feature in a document or text.

TFIDF: Tf-idf is a simple twist on the bag-of-words approach. It stands for term frequency-inverse document frequency. Instead of looking at the raw counts of each word in each document in dataset, tf-idf looks at a normalized count where each word count is divided by the number of documents this word appears in

It is another way to convert textual data to a numeric form. The vector value it yields is the product of these two terms; TF and IDF

Relative term frequency is calculated by:

$$TF\ (t, d) = \frac{number\ of\ times\ term(t)\ appears\ in\ document(d)}{total\ number\ of\ term\ in\ document(d)}$$

And we need to get inverse Document Frequency, which measures how important a word is to differentiate each document by following the calculation as below

$$IDF\ (t, D) = \log(\frac{total\ number\ of\ document(D)}{number\ of\ documents\ with\ the\ term(t) in\ int})$$

Once we have the values of TF and IDF, now we can calculate TFIDF as below

$$TFIDF\ (t, d, D) = TF\ (t, D).IDF\ (t, D)$$

Here if N is the total number of documents in the dataset. The fraction N/ (document……) is what is known as the inverse document frequency. If a word appears in many documents, then its inverse document frequency is close to 1. If a word appears in just a few documents, then the inverse document frequency is much higher.

Alternatively, we can take a log transform instead using the raw inverse document frequency. Logarithm turns 1 into 0, and makes large numbers (those much greater than 1) smaller.

Stop word: There are some words which do not make any significant change in absence of them. Those words are called stop word. Our current step is to remove those words from the document. There are a lot of stop words in English language such as: him, about, ours, those, me, few, how, being, off, again, yourselves, its, once, below, any, yourself, is, from, do, can, until, all, hers, our, just, further, then, above, into, theirs, in, i, who, for, more, each, doing, with, against, o, of, during, as, there, some, are, while, and, only, if, where, were, so, having, these, before, myself, under, very etc.

3.1.3 Training Module

In this module the tfidf matrix of every vector is used in our classifier to train the model. In this way first select the classifier that is best for the model. We prefer logistic regression classifier to make this happen.

Logistic Regression Classifier

Logistic regression is a simple, linear classifier. Due to its simplicity, it's often a good first classifier to make a model. It takes a weight combination of the input features, and passes it through a sigmoid function, which smoothly maps any real number to a number between 0 and 1. The function transforms a real number input x, into a number between 0 and 1. It has one set of parameters w, which represents the slope of the increase around the midpoint, 0.5. The intercept term b denotes the input value where the function output crosses the midpoint. A logistic classifier would predict the positive class if the sigmoid output is greater than 0.5 and the negative class otherwise. By varying w and b, one can control where that change in decision occurs m and how fast the decision should respond to changing input value around that point

Logistic regression is one of the most popular machine learning algorithm for binary classification. This is because it is a simple algorithm that performs very well on a wide range of problem [14]. Logistic regression corresponds to a linear regression where the dependent variable is binary. It is very useful for understanding or predicting the effect of one or more variable on a binary response variable.

The probability for class j with the exception of the last class is [15]:

$$P_j(X_i) = \frac{e^{X_i B_j}}{\left(\sum_{j=1}^{k-1} e^{X_i * B_j}\right) + 1}$$

B: parameter matrix.

K: number of classes.

The last class has probability

$$1 - \sum_{j=1}^{k-1} P_j(X_i) = \frac{1}{\left(\sum_{j=1}^{k-1} e^{\left(X_i * B_j\right)}\right) + 1}$$

In the linear regression classification method the equation is

$$y = bo + b1 * x$$

In this equation if we use a sigmoid function: p=$\frac{1}{1+e^{-y}}$

Then the equation will look like:

$$\ln\left(\frac{p}{1-p}\right) = b0 + b1 * x$$

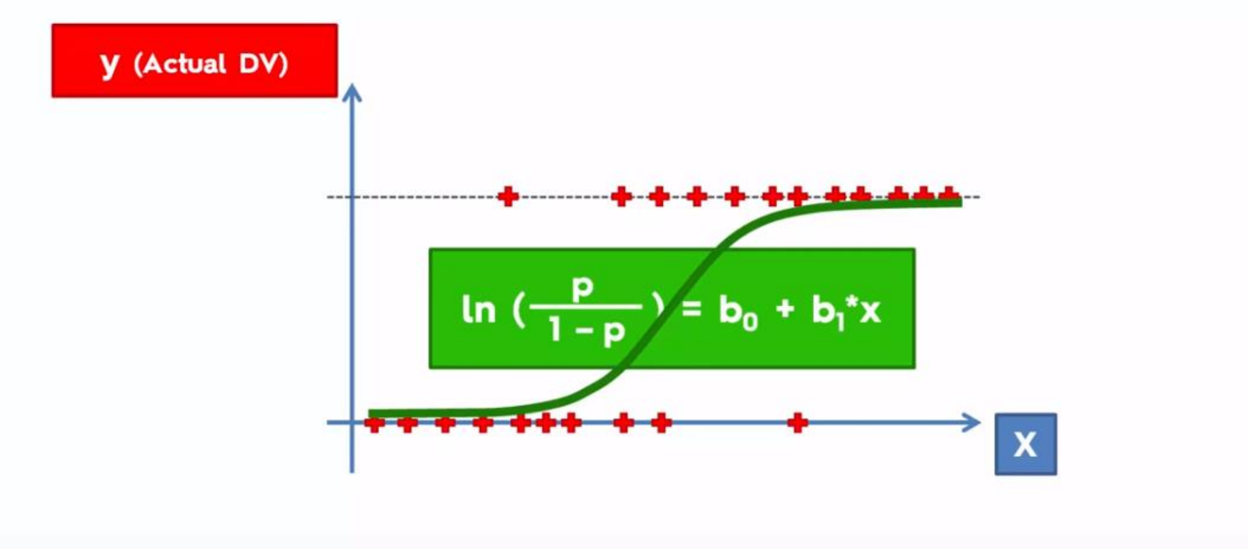


Fig 5: Graphical representation of Logistic Regression Classifier

From this graph we can say that the nonlinear data that don’t have any straight line classifier to differentiate the classes. Thus use the logistic regression to classify the level. Here we can have the probability from the classifier. And from the probability we can predict the polarity of our input

Here in X axis the independent variable the word frequency is plotted and in the Y axis the probability is found.

Here we spit the dataset into training set and testing set for both X and Y axis. And thus have the accuracy of the model by testing the testing leveled data.

3.1.4 Testing In Real-time

In this module the trained model is worked for predicting the conspiracy from the users mail data. Here we have some client those can communicate with email by a mail server. A data crawling method is set to crawl the email data and store it in a database. Then the trained model analyze the data and predict a probability of conspiracy or not.

If the model has the higher probability of conspiracy positive in the body of the mail, then the model give alert to the monitoring body of the company and store the data and those who exchange the mail between them. Thus the verdict of every email is monitored automatically.

3.2 Analytical Representation of the Architecture

This section gives an analytical description of the system architecture given in previous sections. The system architecture above illustrates the internal and external structure of system modules integrated together in one package to form one system. The following subsections provide a brief background overview of the tools used and the implementation details of the different modules of the developed system starting from the back-end to the front-end. The whole system was developed Windows operating system, PhpStorm IDE and PyCharm IDE platforms

3.2.1 Labelling the Email Data

We have labeled the email data by Green data and Red data. We collect the Green email from the Enron dataset. And the Red dataset are collected from the practical field.

3.2.2 Clean the Data

After data is labeled then we have the raw email data. We cannot use these data to classify or train. So, we have cleaned the data before using them in classifier or training. We have performed several cleaning like removing URL, removing stop words, removing multiple spacing etc.

Algorithm 3.1: Cleaning raw email

Input: raw email

Require: clean the raw email

```
1. Begin
 2. Remove url from raw email data
 3.  Convert the raw email into lowercase form
```

```
Search for contracted form in email body
if contracted form found then
    Replace it with long form
Search for stop words in data
if stop words found then
    Remove the stop words
End
```

3.2.3 Process the data

After cleaning data from the garbage data then the email data is ready to be processed by vectorization. Here we will use the TFIDF vectorization process to split and calculate the importance of any word in the dataset.

Algorithm 3.2: Process the cleaned email

Input: cleaned email

Require: process the cleaned email

```
Begin
Remove stop words from raw tweets
Convert the raw email into lowercase form
Tokenize the tweets
Calculate the TFIDF matrix
End
```

3.2.4 Train the Model

To predict the classes of the email we need a mathematical model that can specify the class of the email based on their features. We have used Logistic Regression classifier algorithm. There are some more algorithm for classification. We use Logistic Regression as it is a probabilistic method and we have a small amount of training dataset

3.2.4.1 Logistic Regression

Algorithm 3.3: Logistic Regression learning algorithm

Inputs: Training data, x

Require: Train model to classify

```
Begin
Initialize w
for i=1 to n do
    $z(i)=\sum w(i)*x(i)$
end for
for j = 0 to d do
    for i = 1 to n do
        $\theta(j)$ = SOFT-MAX(z(i))
    end for
end for
End
```

3.2.5 Collecting the Mail Data in Real Time

In the real time classification process we use a crawling algorithm and store it in a database. We crawl the data and the communicating employee’s name. Then it stores in another database for further analyzation.

Algorithm 3.4: Collect the email data from the profile

Inputs: Automated process

Require: Take the data to another database

```
Begin
Access the storing database of the email
if the email is not still taken
    take the email
End
```

3.2.6 Generating Output

By using the model that we have built in the previous steps, we can classify email body. The classification result is 0 or 1. According to this result we can show the type of the email.

Algorithm 3.5: Classification of real-time email

Inputs: model file

Require: Classification of the email

```
Begin
classifier = load(model)
for each email in the emaildata table in database test
    take the email body
```

```
type = classifier.predict(email body)
if type = 0 then
    result = “It is not infected”
else if type = 1 then
    result = “Infected”
    store the email with the sender and receiver name in another database
    show the result
End
```

3.3 Complexity Analysis

Our system has three parts keeping the issue of time in concern. The time complexity of our system may be described as follows:

Complexity of Email Cleaning

Let, n is the number of letter in a email, m is the number of email in the dataset So time require to clean all the email are O (nm).

Complexity of storing the database of Email

Let we have n number of email in any email database. So it will take to crawl the data from the dataset to another dataset. It will take the complexity of O (n).

Complexity of Predicting an Email

Let, n is the number of word in a email, C is the number of words in the feature vector. So time require to predict an email is O (nC).