Tribhuvan University

Institute of Engineering, Pulchowk Campus

Final Report

Web Spam Detection based on Artificial Neural Network

Submitted by:
Ajaya Puri
072-MSCS-651

Submitted to:
Department of Electronics and Computer Engineering
May, 2017

Final Report
On

Web Spam Detection based on Artificial Neural Network

Submitted by:
Ajaya Puri
072-MSCS-651

Submitted to:
Department of Electronics and Computer Engineering
May, 2017

ABSTRACT

Web spamming refers to actions intended to mislead search engines into ranking some pages higher than they deserve. Search engines are the gateways to the web that is why some people try to mislead search engines, so that their pages would rank high in search results, as a consequence genuine web page got lower rank from search engine than those artificially boosted webpages.

Basically spammer use two approach for mislead result from search engine. The first category includes the boosting techniques, i.e., methods through which one seeks to achieve high relevance and/or importance for some pages. The second category includes hiding techniques, methods that by themselves do not influence the search engine's ranking algorithms, but that are used to hide the adopted boosting techniques from the eyes of human web users.

For evaluation and experiments, machine learning approach is used here. Multilayer perceptron neural network is applied as learner. Overall experiment is performed on WEBSPAM-UK-2007 publicly available dataset.

Keywords

Web spam detection, content spam, link spam, WEBSAPM-UK-2007, Neural Network, Gradient Descent

TABLE OF CONTENTS

List of Figures

List of Tables

List of Abbreviations

ANN	Artificial Neural Network
CSV	Comma Separated Value
GD	Gradient Decent
HTML	Hyper Text Markup Language
IDF	Inverse Document Frequency
LM	Levenberg-Marquardt
MLP	Multi-layer Perceptron
NN	Neural Network
PR	Page Rank
SOM	Self-organizing map
TR	Term Frequency
UK	United Kingdom
URL	Uniform Resource Locator

CHAPTER ONE

1.1 Introduction

Internet is the source of information for all. Search engine is the gateway to the internet. The objective of a search engine is to provide high quality results by correctly identifying all web pages that are relevant for a specific query, and presenting the user with some of the most important of those relevant pages [1]. Today is mid of April, 2017, worldwide around one billion websites are registered with more than four billion webpages and three and half billions of user are there. An intelligent community is standing here and are trying to bypass the rule that normal search engine follow. The overall approach is to deceive the correctness of search result and rank the webpage in higher rank than actually they are.

Web spamming refers to actions intended to mislead search engines into ranking some pages higher than they deserve [1].

There are several malicious methods that try to circumvent the search engines by manipulating the relevance of web pages. This deteriorates the search results, leaves users frustrated and exposes them to inappropriate and insecure content. Such technique is known as web spamming [2].

Recent trend of web spamming is done by two method, first content spamming and second link spamming [3]. In Content based spamming, spammers can attach their unsolicited content (i.e. spam) to pages, and has several variants like title spamming, body spamming, Meta tag spamming, anchor-text spamming, URL spamming.

In link spamming technique, spammers tend to insert links between pages that are present for reasons other than merit. Link spam takes advantage of link-based ranking algorithms, such as Google's Page Rank algorithm, which gives a higher ranking to a website that is cited by other high ranked websites [4].

Web spam identification method being used are supervised as well as unsupervised learning method. Traditionally support vector machine, k-NN, Naïve network had been used successfully. The main drawback of such an approach is that it works well only for small datasets and a limited number of class labels. So why I am interested in artificial neural network for performing this task. ANN is well suited for both supervised and unsupervised learning method.

1.2 Problem definition

Web spam has gained a lot of attention as Web search engines are the ultimate tool to retrieve information. Web spam detection is a necessary due to its devastation towards Web search engines and global cost of billion dollars annually.

1.3 Objective

Prime objective of this project work is to measure webpages as a spam or non-spam using neural network.

Scope of Work

This project will have application in following fields:

Web Indexing by Search Engine
Web spam detection is a classification problem, and search engines use machine learning algorithms to decide whether or not a page is spam. Search engine may also analyses for Web Search Relevance Ranking.

Firewall Protection
Firewall could use machine learning approach if any webpage is spam or non-spam to counter internet phishing.

1.4 Literature review

Web spam detection methods and algorithm are mainly categorized in two section. Supervised learning technique and unsupervised technique [2, 4,5].
Hassan, Hmeidi involved Naïve Bayes classifier in discovering non-required web pages. In training phase, a user-oriented preparation was performed, wherein, the web pages reside in the web server are classified into spam or non-spam based on user's feedback and the automated classification by the filter. The experimental results showed that Naïve Bayes classifier provides on average accuracy equal to 80.2% [4]

Silva, Almeida, and Yamakami worked in spam detection with different artificial neural network. Basically they worked in multilayer perceptron and self-organizing map (SOM) for with different training algorithm. They found self-organizing map is inferior to multilayer perceptron for all simulation. The simulation was performed by training the ANN with gradient decent (GD) and Levenberg-Marquardt (LM) algorithm. To address the algorithms performance, they used a random sub sampling validation, which is also known as Monte Carlo cross-validation. They had used three sets of 8,487 feature vectors employed to discriminate the hosts as spam or ham. Their works shows that best result was acquired when all features of content and link are used. The average accuracy level of their work was nearly 86.6% for GD+MLP, 88.2% for LM+MLP and 80.6% for SOM network. Their working dataset was WEBSPAM UK-2006.They implemented all the MLPs with a single hidden layer and with one neuron in the output layer [2].

Roul, Asthana, Shah and Parikh used a combined approach of content and link-based techniques to detect the spam Web pages. They used a set .html Web as the input to the procedure and the output was detection of each page as spam or non-spam. For every page, they measured term density test, POS ratio test and link-based test. WEBSPAM-UK2006 dataset had been used to obtain the appropriate threshold values [5].

CHAPTER TWO

2.1 Web Spam Taxonomy

Content Spam

The content spam is probably the first and most widespread form of web spam. It is so widespread because of the fact that search engines use information retrieval models based on a page content to rank web pages [3].Thus, spammers analyze the weaknesses of these models and exploit them.

Link Spam

Link spam are links between pages that are specifically set up to take advantage of link-based ranking algorithms such as Google's PageRank (PR).

Cloaking

Cloaking is the way to provide different versions of a page to crawlers that is substantially different from the search engine description. For example, delivering pornographic content cloaked within non-pornographic search results [3].

Redirection

Process of directing the browser to another URL as soon as the page is loaded. This way the page still gets indexed by the search engine, but the user will not ever see it—pages with redirection act as intermediates (or proxies, doorways) for the ultimate targets, which spammers try to serve to a user reaching their sites through search engines[1].

Spammer use two techniques associated to trick the search engine result.

1. Boosting Technique
2. Hiding Technique

Boosting technique use

1 Term

2 Link

2.2 Term Boosting

Search engines consider where on a web page query terms occurs. Each type of location is called a field. The common text fields for a page are the document body, the title, the meta-tags in the HTML header, and page p's URL [1]. The algorithms used by search engines to rank web pages based on their text fields use various forms of the fundamental TFIDF metric used in information retrieval. The TFIDF score of a page p with respect to a query q is then computed over all common terms t[1]:

$$\mathrm{TFIDF}(p, q) = \sum_{t \in p, t \in q} \mathrm{TF}(t).\mathrm{IDF}(t)$$

Term Frequency indicate how many times a term repeated in a document. The inverse document frequency IDF (t) of a term t is related to the number of documents in the collection that contain t.

With TFIDF scores in mind, spammers can have two goals: either to make a page relevant for a large number of queries by including a large number of distinct terms in a document or make a page very relevant for a specific query by repeating some "targeted" terms.

Term spamming techniques can be grouped based on the text field in which the spamming occurs.

Body spam. In this case, the spam terms are included in the document body.

Title spam. Today's search engines usually give a higher weight to terms that appear in the title of a document. Hence, it makes sense to include the spam terms in the document title.

Meta tag spam. The HTML meta tags that appear in the document header have always been the target of spamming.
<meta name="keywords" content="buy, cheap,cameras, lens, accessories, nikon, canon">

Anchor text spam: search engines assign higher weight to anchor text terms, as they are supposed to offer a summary of the pointed document
free, great deals, cheap, inexpensive, cheap, free

URL spam: Some search engines also break down the URL of a page into a set of terms that are used to determine the relevance of the page. Spam URLs look like
buy-canon-rebel-20d-lens-case.camerasx.com,
buy-nikon-d100-d70-lens-case.camerasx.com,

2.3 Link Boosting

Search engines also rely on link information to determine the importance of web pages. Therefore, spammers often create link structures that they hope would increase the importance of one or more of their pages. For a spammer, there are three types of pages on the web:

Inaccessible pages are those that a spammer cannot modify. These are the pages out of reach; the spammer cannot influence their outgoing links.

Accessible pages are maintained by others but can still be modified in a limited way by a spammer. For example, a spammer may be able to post a comment to a blog entry, and that comment may contain a link to a spam site.

Own pages are maintained by the spammer, who thus has full control over their contents.

We could group link spamming techniques based on whether they add numerous outgoing links to popular pages or they gather many incoming links to a single target p.

Outgoing links. A spammer might manually add a number of outgoing links to well-known pages, hoping to increase the page's hub score.

Incoming links. In order to accumulate a number of incoming links to a single target page, a spammer might adopt some of the following strategies:
Create a honey pot, a set of pages that provide some useful resource (e.g., copies of some Unix documentation pages), but that also have (hidden) links to the target spam page(s).

Post links on blogs, wikis.
Spammers may include URLs to their spam pages as part of the seemingly innocent comments/messages they post.
Participate in link exchange. Often times, a group of spammers set up a link exchange structure, so that their sites point to each other.

Create own spam farm. These days' spammers can control a large number of sites and create arbitrary link structures that would boost the ranking of some target pages.

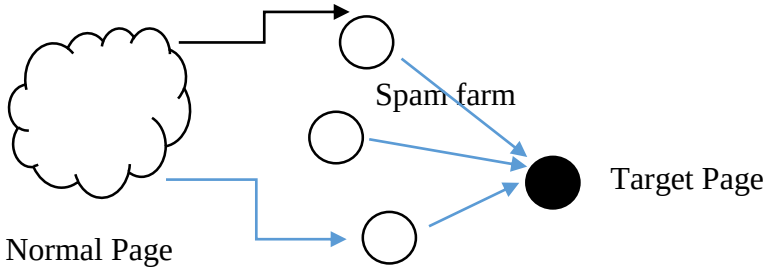


Figure 1 Spam Farming

2.4 Content Hiding
Spam terms or links on a page can be made invisible when the browser renders the page.

```
<body bgcolor="white">
<font color=“white”>hidden text</font>
. . .
</body>
```

2.5 Cloaking
If spammers can clearly identify web crawler clients, they can adopt the following strategy, called cloaking: given a URL, spam web servers return one specific HTML document to a regular web browser, while they return a different document to a web crawl

CHAPTER THREE

3.1 Artificial Neural Network (ANN)

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired from biological nervous systems. It has large number of highly interconnected processing elements, each single unit is termed as neuron. These neurons work in parallel to solve specific problems.

3.2 Single Neuron as Computing Node

A single neuron transforms the weighted sum of its inputs via a function, usually non-linear, into an activation level, alternatively called the output.

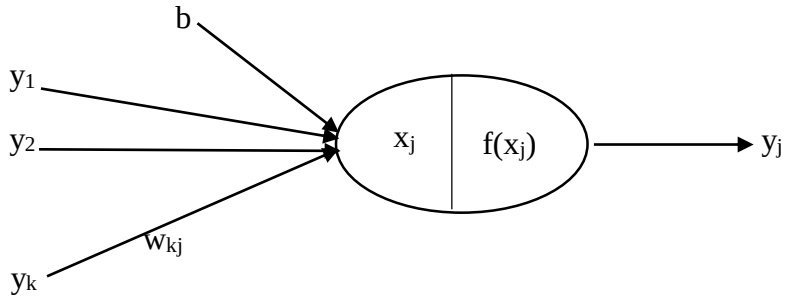


Figure 2 Single Neuron

Where $y_1, y_2, \ldots, y_k$ is input for given j^{th} neuron, b is bias input and y_j is output of j^{th} neuron and y_{kj} is synaptic weight form k^{th} neuron to $j^{th.}$

Mathematically,

$x_j = \sum w_{kj}.y_k + b$ for all input of k^{th} node to j^{th} node

$y_j = f(x_j)$

The neuron takes a weighted sum (x_j) of the inputs (y_k), and passes it through the activation function f(.) to produce the output y_j .

The f(.) is nonlinear activation function typically chosen to be the sigmoidal function.

$$f(x) = \frac{1}{1+e^{-x}} \quad \text{Eq. 1}$$

an first order derivative of Eq. 1 is

$$f'(x) = f(x).(1 - f(x)) \quad \text{Eq. 2}$$

The reason behind choosing the activationn fuction sigmaoidal is that it has finite limits at negative infinity to infinity, most often going either from 0 to 1 or from −1 to 1. Sigmoidal funciton looks like,

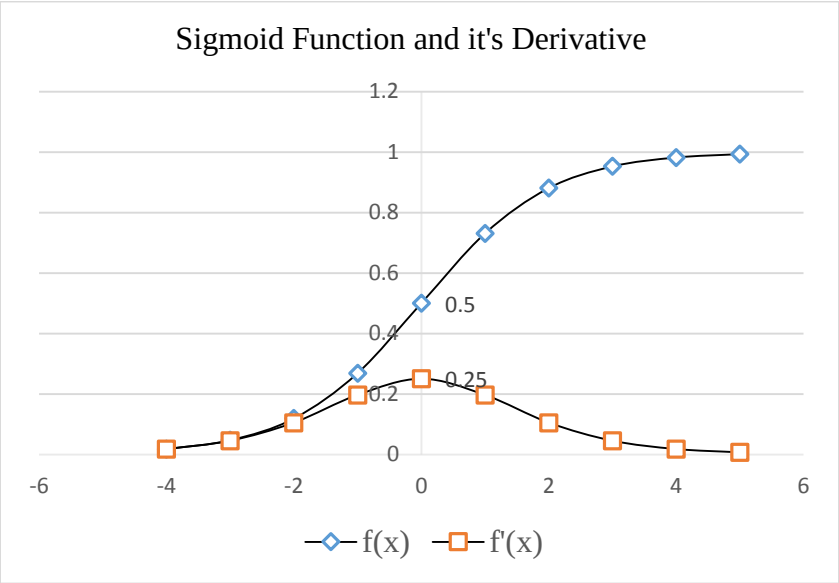


Figure 3 Sigmoid function

3.2 Multilayer Perceptron Neural Network

Elementary neurons are combined together to form a parallel computing unit, called multilayer perceptron neuron network. The computing units are categorized into three layer.

Input Layer: Collection of computing node which receive external data.

Output Layer: Final node which send data as output.

Hidden Layer: Computing nodes which receive output of previous layer's as input and pass to the network.

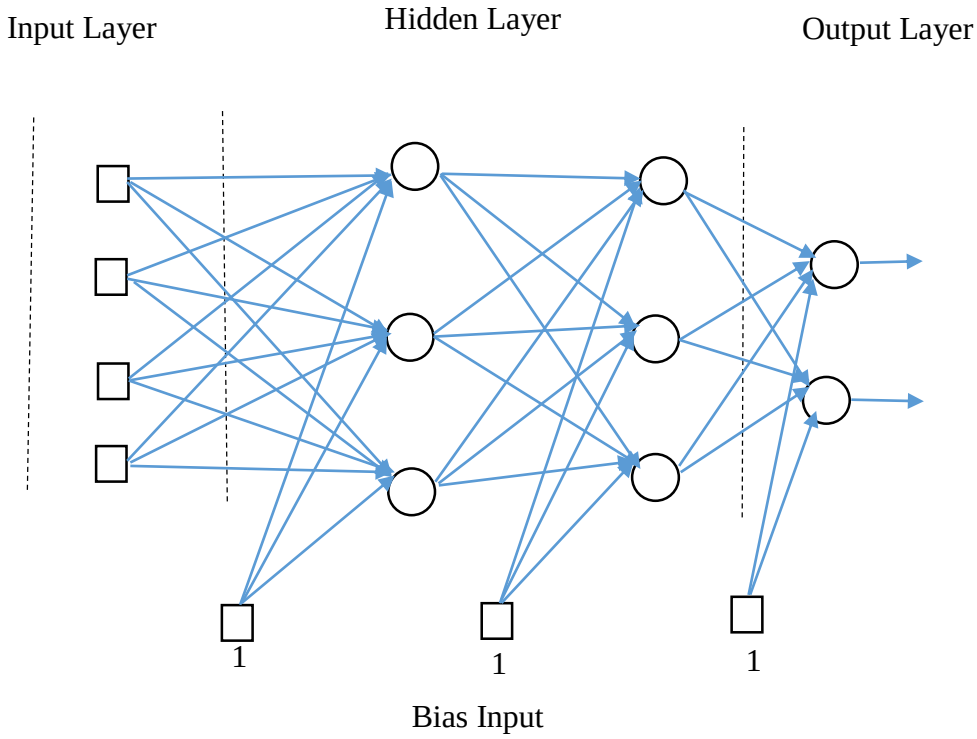


Figure 4 Multi-Layer Perceptron Network

3.3 Learning paradigms

Basically done through supervised learning, unsupervised learning and reinforcement learning.

In supervised learning, we will have set of input data and corresponding output for those datasets. A supervised learning algorithm analyzes the training data and produces a concluded function, which can be used for mapping new dataset.

In unsupervised learning, learner will have set of data but no output is available. Learner self-evaluate the output from dataset.

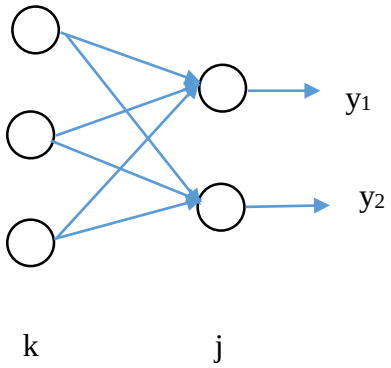
In reinforcement learning, learner will neither have dataset nor its corresponding output. Paradigm needs external environment to reward learning skill of agent. A reinforcement learning agent interacts with its environment in discrete time steps. At each time t, the agent receives an observation. It then chooses an action from the set of actions available, which is subsequently sent to the environment.

3.4 Back propagation in MLP Neural Network

Supervised learning methods use back propagation algorithm which can be summarized in two stages: forward and backward. In the forward stage, the signal propagates through the network,

layer by layer. In backward stage, the derivation of the back propagation algorithm is performed starting from the output layer.

Learning occurs in MLP by adjusting synaptic weights after each piece of data is processed. Error measured in output layer propagated to previous layer in order to train network correctly. Error value measured at output layer and hidden layer is calculated differently.



The error measured at output node is the difference between the actual output (y_j) and the targeted output (t_j) for that node. The overall error at output node is summation squared error (negative and positive errors not to cancel each other out) for each computing node.

$$E = \frac{1}{2}\sum_{j=1}^{N}(t_j - y_j)^2 \qquad \text{Eq. 1}$$

Where, N is total number of computing node at output layer.

Change in weight connecting a node in layer k to a node in layer j is

$$\Delta w_{kj} = -\alpha \frac{\partial E}{\partial w_{kj}} \qquad \text{Eq. 2}$$

Here is α free parameter called "learning rate" that we set prior to training. Negative sign indicates that weight changes are in the direction of decrease in error. Expanding partial derivative:

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial x_j}\frac{\partial x_j}{\partial w_{kj}}$$

where $x_j = \sum w_{kj}.y_k + b$

Therfore, $\frac{\partial x_j}{\partial w_{kj}} = y_k$

and say $\delta_j = -\frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial x_j}$ called error term, thus

$$\frac{\partial E}{\partial w_{kj}} = -\delta_j y_k \qquad \text{Eq 3}$$

we have $y_j = f(x_j) = \frac{1}{1+e^{-x_j}}$ is sigmoidal function, derivative is

$$\frac{\partial y_j}{\partial x_j} = y_j(1 - y_j) \quad \text{Eq. 4}$$

For output layer from equation 1

$$\frac{\partial E}{\partial y_j} = -(t_j - y_j)$$

$$\frac{\partial E}{\partial w_{kj}} = -(t_j - y_j).y_j(1 - y_j).y_k$$

and $\delta_j = (t_j - y_j).y_j(1 - y_j)$ Eq. 5

and if j hidden layer

$$\delta_j = (\sum \delta_i w_{ji}).y_j(1 - y_j) \quad \text{Eq. 6}$$

Combining equation 2 and 3

$$\Delta w_{kj} = \alpha \delta_j y_k \quad \text{Eq. 7}$$

Finally update the weight as

$$\Delta {w_{kj}}^n = \alpha \delta_j y_k + \eta \Delta {w_{kj}}^{n-1} \quad \text{Eq.8}$$

Here, n and n-1 indicate iteration number.

α is learning rate, η is the momentum, a real value on the interval (0,1].

3.5 Cross Validation in Machine Learning Algorithms

A learning algorithm is said to over-fit if it is more accurate in fitting known data but less accurate in predicting new data. Over-fitting is likely seen in cases where learning was performed too long or where training examples are rare, causing the learner to adjust to very specific random features of the training data. In this process of over-fitting, the performance on the training examples still increases while the performance on unseen data becomes worse.

Cross Validation (CV) is a technique for determining how the results of a statistical analysis will generalize to an unseen dataset. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.

The goal of cross validation is to define a dataset to examine the model in the training phase (i.e., the validation dataset), in order to limit problems like over-fitting, give an insight on how the model will generalize to an unseen dataset.

Basically there are four type of cross validation- Holdout CV, K-fold CV, Leave one out CV and Bootstrap CV.

In holdout cross validation the available data sample are partitioned into two subset and partition is random.

Training Sample (60%-70%)	Testing Sample (40%-30%)

The first sample is training sample typically size of about 70% over which we train our system. The second one is testing sample of remaining size where model is validated how well it has generalize. We compare error parameter on both sample set and if they are similar to each other, we can conclude our model is well parameterized. In this approach sample bias is seen. This is the case where we do not have well known pattern for dividing the dataset into training and testing. So that out model could perform well for training sample where as it fails in testing sample.

In k-fold cross validation, the available sample size are partitioned into k sub sample of equal size. In one iteration of run, k-1 sample are treated as training sample and one sample set is validation sample. The iteration goes to k times by changing validation sample to next other than previous validation sample. The benefit of this validation method is all sample are trained and validated so model do not suffer from sample bias.

In leave one out cross validation, the sample are divided into N subsample, where N is the total sample size. This is similar to k-fold cross validation, the one difference is fold value equal to number of dataset sample.

In bootstrap cross validation, bootstrap sample set that has equal dataset as in original set is formed. But replacement of any dataset to original may occurs in bootstrap samples. For example let us say original dataset $D = \{d_1, d_2, d_3\}$, then bootstrap sample may be

$B_1 = \{d_1, d_1, d_3\}$ $B_2 = \{d_1, d_1, d_2\}$ $B_1 = \{d_1, d_2, d_2\}$

Note that bootstrap sample B_1 contain dataset from D but d_2 is replaced with d_1.

CHAPTER FOUR

4.1 Webspam-UK-2007 datasets

This project work has been done in publicly available dataset WEBSPAM-UK-2007. The base data is a set of 105,896,555 pages in 114,529 hosts in the .UK domain. The number of feature sets are tabulated as

Notation	Feature Set	No. of Features
A	Content-based Features	24
B	Full Content-based Features	96
C	Link-based Features	41
D	Transformed Link-based Features	138

Among those I used content-based and link based feature set. Feature used in my project for experiment is

SN	Content Based Features (home page)
1	Number of words
2	Number of words in the title
3	Average word length
4	Fraction of anchor text
5	Fraction of visible text
6	Compression rate of the homepage
7	Top 100 corpus precision

SN	Link based Features
1	assortativity_hp
2	indegree_hp
3	outdegree_hp
4	pagerank_hp
5	reciprocity_hp
6	trustrank_hp
7	truncatedpagerank_1_hp

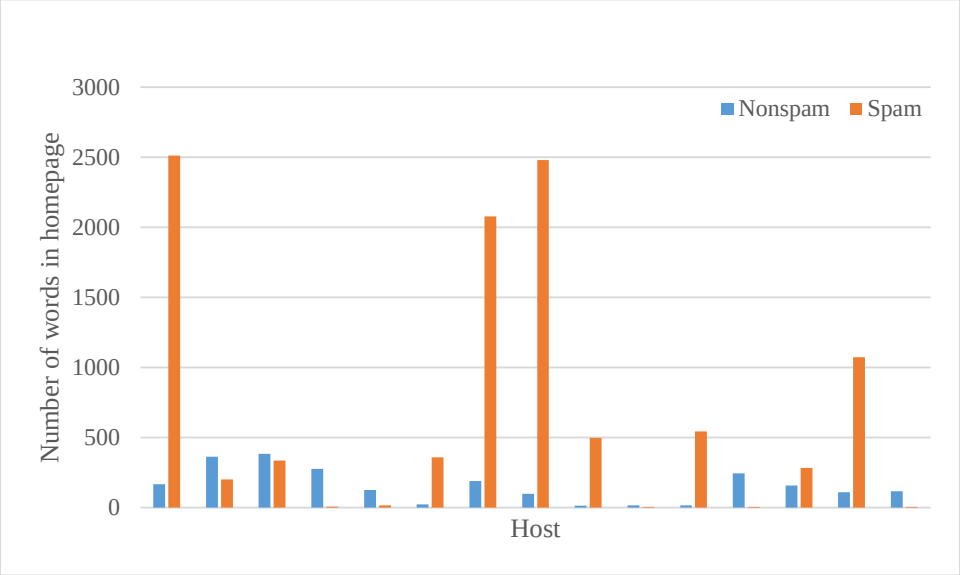
Sample for one host looks like

hostid	HST_1	HST_2	HST_3	HST_4	HST_5	HST_6	HST_7	Remark
4	62	8	4.967742	0.016129	0.033315	1.761905	0.112903	nonspam
112	85	4	5.376471	0.058824	0.024927	1.862543	0.058824	spam
822	49	14	4.897959	0	0.014348	1.7	0.142857	undecided

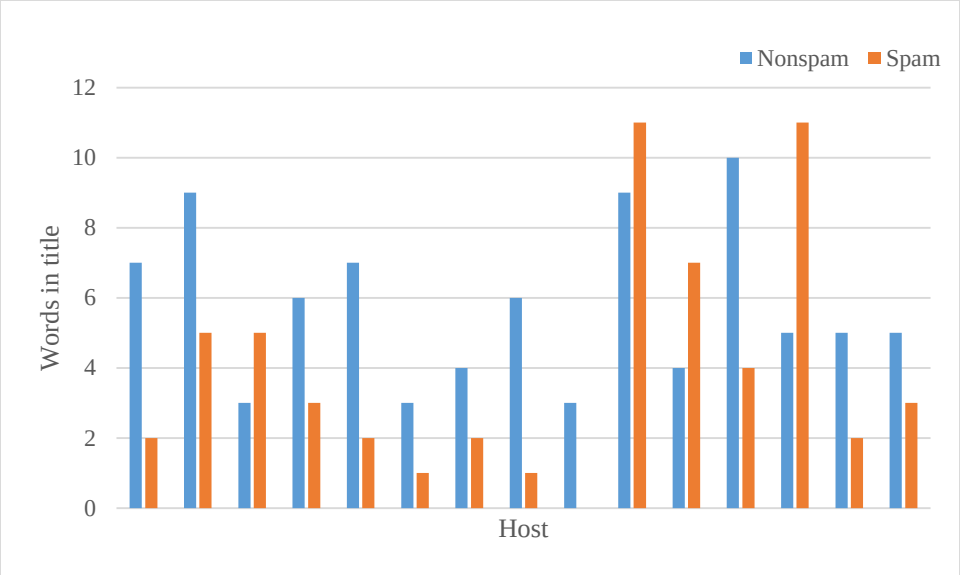
Note HST_1 to HST_7 represent content based feature arranged serially in above feature set table. E.g. HST_1 represent number of words in the page.

#hostid	assortativity_hp	indegree_hp	outdegree_hp	pagerank_hp	reciprocity_hp	trustrank_hp	truncatedpagerank_1_hp	Remarks
4	0.001186	4	1	2.20E-09	0	5.94E-10	8.48E-10	nonspam
112	0.613757	24	5	3.83E-08	1	9.57E-09	4.09E-08	spam
223	166.2956	20118	14	2.65E-06	0.866667	1.43E-05	3.13E-06	undecided

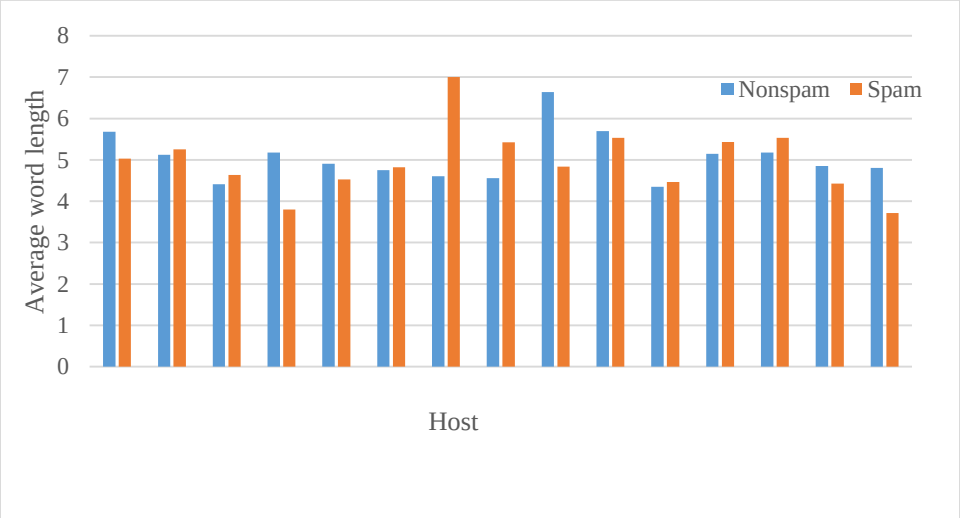
4.2 Sample Data Analysis



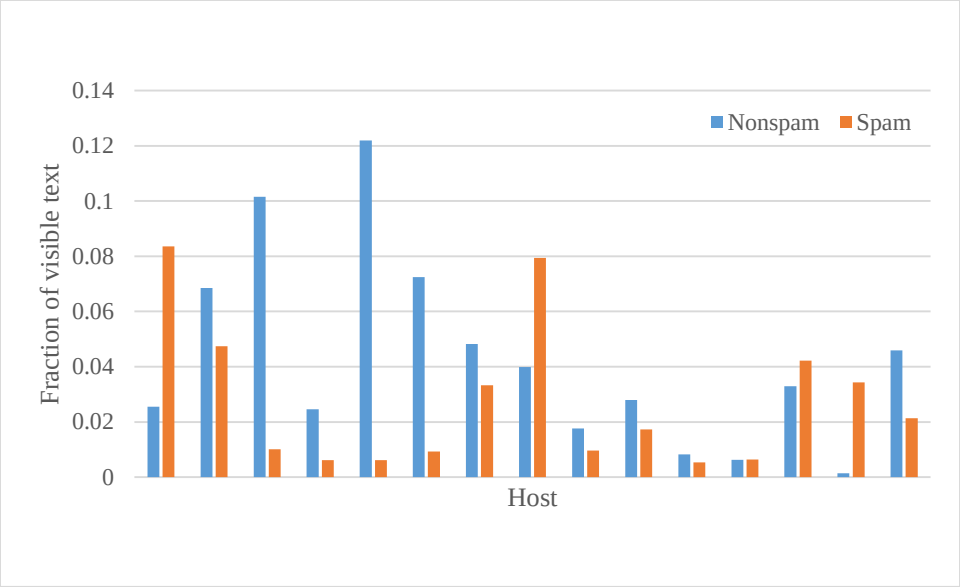
As seen in bar diagram, the webpages with high number of word in homepage are likely to be spam page as it could embed more suspicious words that lead finally as spam pages.



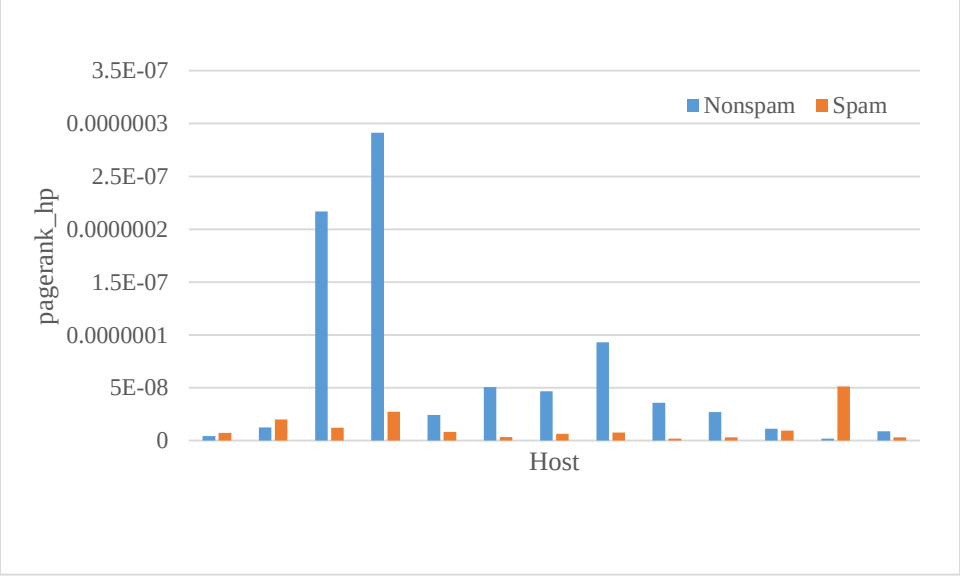
Similar to number of words in homepage, suspicious words could be in title page and final goal of spammer could be misled the search engines. We could see spam webpages with high diversity.



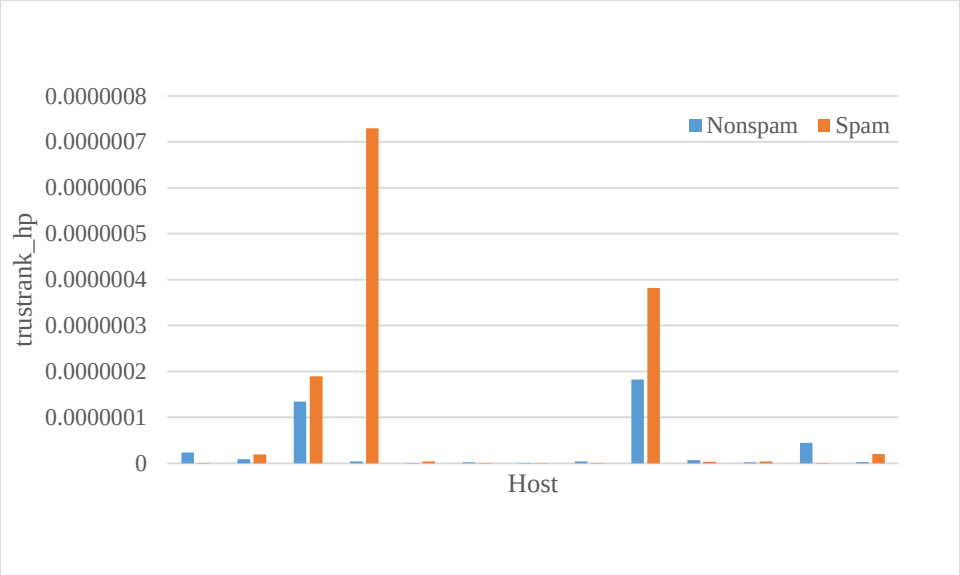
Average word length in homepages are seen little higher in case of spam page as compared to nonspam pages.



Fraction of visible text represent ratio of visible text to total text. High values indicate higher portion of text are visible to end user. So the possibility to hide suspicious link and words are reduced.



Pagerank use incoming link information to measure the importance of the webpages. The pages which receive high link will lead high value of pagerank. Pagerank also indicate the page which receive income link from higher important page will be high. So spammer has no role to give higher importance to targeted pages, but they could use other secondary approach, this effect could clearly seen in above diagram where nonspam page has higher pagerank values.



Basic idea of trustrank is, the genuine page rarely outlink to suspicious pages. Trustrank seeks to combat spam by filtering the web based upon reliability.

CHAPTER FIVE

5.1 Methodology

5.1.1 Functional Block Diagram

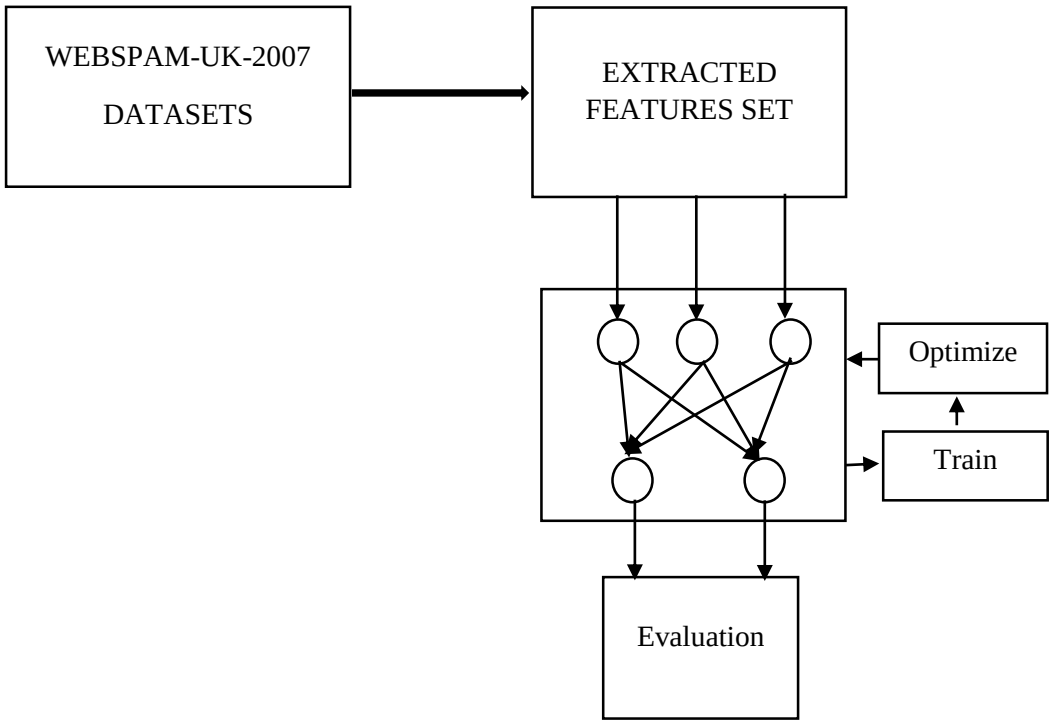


Figure 5 Functional Diagram

5.1.2 Pseudo code

initialize network weights with small random values

define transfer function $y_j = \frac{1}{1+e^{-x_j}}$, learning rate α,

do

forEach training row

Compute output, y_j

Read target, t_j

Compute error, δ_j

at output $\delta_j = (t_j - y_j).y_j(1 - y_j)$

at hidden $\delta_j = (\sum \delta_i w_{ji}).y_j(1 - y_j)$

update network weights from output layer(i) to first hidden layer

$\Delta {w_{kj}}^n = \alpha \delta_j y_k + \eta \Delta {w_{kj}}^{n-1}$

until all examples classified correctly

return

Explanation

Overall project work is separated in two phase, data preprocessing data and execution phase.

1. Data preprocessing phase

In data preprocessing phase, WEBSPAM-UK-2007 has pre-computed feature as described in chapter. This data is need to be combined with spam label dataset, which has basically spamicity label as well as measure.

5 nonspam 0.000000 j24:N,j32:N
112 spam 1.000000 j31:S
223 undecided 0.500000 j13:B,j28:B
1223 undecided - j6:U,j37:U

First three parameter indicate hostid, classification label and spamicity measure. Data is classified into three label as spam, nonspam and undecided. In my work, I have to join this classification label to featured dataset. Finally I got my working data as

[[62,8,4.967741935,0.016129032,0.033315422,1.761904762,0.112903226,1],
[436,1,4.610091743,0.020642202,0.009361848,2.180035651,0.231651376,2],
[234,8,5.226495726,0.098290598,0.028928174,2.258914729,0.11965812,3]]

The last number in each list indicate numerical representation of classification label. Here 1 is for nonspam, 2 is for spam and 3 is used for undecided dataset. All working dataset is formed in .cvs format.

2. Execution Phase

In this, real working of dataset are fed into network and classification is done. Process is described shortly herewith.

Neural network is designed with 5 neuron in hidden layer, 3 with output layer, learning rate is computed as .4. These parameters were found with a little trial and error.

The learning of network is basically done by repeated call of forward and backward pass.

In forward pass, all computing node are activated by preceding node's output, error for output layer and hidden layer are computed as

For output layer, j is output layer

$$\delta_j = (t_j - y_j).y_j(1 - y_j)$$

and if j hidden layer

$$\delta_j = (\sum \delta_i w_{ji}).y_j(1 - y_j)$$

Finally update the weight as

$$\Delta w_{kj}{}^{n} = \alpha \delta_j y_k + \eta \Delta w_{kj}{}^{n-1}$$

Experiment was done the algorithm using k-fold cross-validation with 5 folds. Dataset used here is 4104. This indicate that 4104/5(821) records will be in each fold. The neural network parameter used in this project work are
Number of hidden Layer: 2
Neuron in hidden layers: 5
Neuron in Output Layer: 3

CHAPTER SIX

6.1 Experiments and Results

Experiment was done in data set of 41XX plus individually on content based and linked based features sets. The Summary of Experiment is shown as,

Experiment Performed on	Accuracy Measure in 1
Content Based Features Set	0.8870
Linked Based Features Set	.8832
Content, Linked	.8897

Table 1 Experimental Result

The graphical analysis of each set of features are depicted as,

Content Based Experimental Result

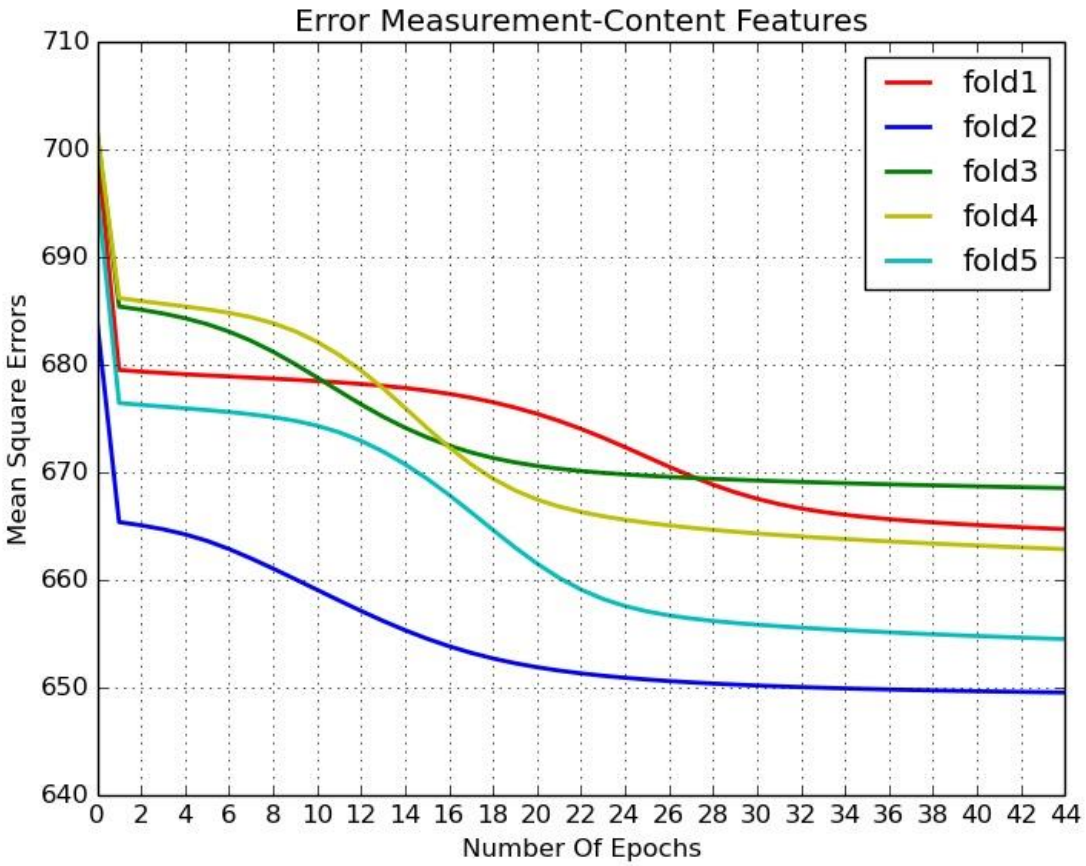


Figure 6 Error graph in Content Based Features

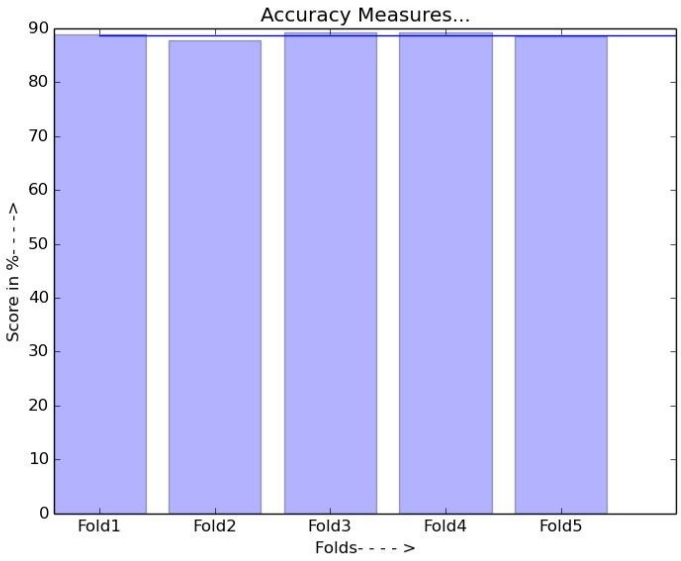


Figure 7 Performance Measure bar-diagram

Linked Based Result

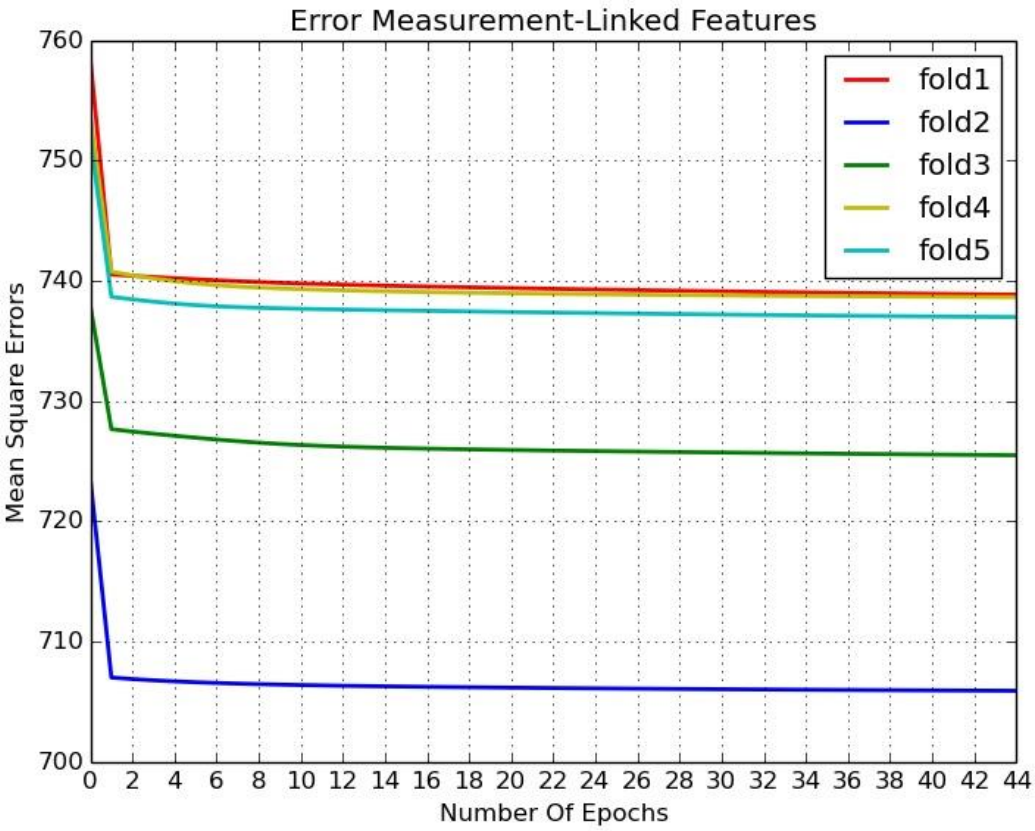


Figure 8 Error graph in Linked Based Features

Neural Network Performance was validated through k-fold cross validation, so fold error and accuracy can be seen in above diagram. In conclusion, this experiment has shown accuracy of about 88 percent.

As I used gradient decent algorithm for error correcting method, it has limitation of being stopped at local minima. Further work could be done on other algorithm.

6.2 Conclusion and Future works

From this project work, we saw the overall accuracy found about 88 percent in all three scenario, content based, linked based and content plus linked based features sets.

The parameters used in this experiment is basically related to homepage content, for content based feature sets. The experiment could be extend to other feature set. This could obviously increase the performance measures.

This work could be done and analyzed in other machine learning algorithms and comparative study could be done. In this work, supervised learning approach was used, same work could be in unsupervised approach, and thus comparative study could achieved.

REFERENCES

1. Zoltan Gyongyi,Hector Garcia-Molina,"Web Spam Taxonomy",Stanford University, First international workshop on adversarial information retrieval on the web (AIRWeb 2005)
2. Silva R.M., Almeida T.A., Yamakami A. (2012) Towards Web Spam Filtering with Neural-Based Approaches. In: Pavón J., Duque-Méndez N.D., Fuentes-Fernández R. (eds) Advances in Artificial Intelligence – IBERAMIA 2012. IBERAMIA 2012. Lecture Notes in Computer Science, vol 7637. Springer, Berlin, Heidelberg
3. Nikita Spirin,Jiawei Han,"Survey on Web Spam Detection: Principles and Algorithms",University of Illinois
4. Hassan Najadat, Ismail Hmeidi, "Web Spam Detection Using Machine Learning in Specific Domain Features", Journal of Information Assurance and Security 3 (2008) 220-229
5. Rajendra Kumar Roul, Shubham Rohan Rsthana, Mit shah and Dhruvesh Parikh, "Detection of spam web page using content and link-based techniques", Sadhana Vol. 41, No. 2, February 2016, pp. 193–202

BIBLIOGRAPHY

1. http://chato.cl/webspam/datasets/uk2007/
2. http://www.webspam.org/
3. http://chato.cl/webspam/
4. https://www.google.com/webmasters/tools/spamreport