DOI: 10.21917/ijsc.2013.0075

REPTREE CLASSIFIER FOR IDENTIFYING LINK SPAM IN WEB SEARCH ENGINES

S.K. Jayanthi[1] and S. Sasikala[2]
[1]*Department of Computer Science, Vellalar College for Women, India*
E-mail: jayanthiskp@gmail.com
[2]*Department of Computer Science, KSR College of Arts and Science, India*
E-mail: sasi_sss123@rediff.com

Abstract

Search Engines are used for retrieving the information from the web. Most of the times, the importance is laid on top 10 results sometimes it may shrink as top 5, because of the time constraint and reliability on the search engines. Users believe that top 10 or 5 of total results are more relevant. Here comes the problem of spamdexing. It is a method to deceive the search result quality. Falsified metrics such as inserting enormous amount of keywords or links in website may take that website to the top 10 or 5 positions. This paper proposes a classifier based on the Reptree (Regression tree representative). As an initial step Link-based features such as neighbors, pagerank, truncated pagerank, trustrank and assortativity related attributes are inferred. Based on this features, tree is constructed. The tree uses the feature inference to differentiate spam sites from legitimate sites. WEBSPAM-UK-2007 dataset is taken as a base. It is preprocessed and converted into five datasets FEATA, FEATB, FEATC, FEATD and FEATE. Only link based features are taken for experiments. This paper focus on link spam alone. Finally a representative tree is created which will more precisely classify the web spam entries. Results are given. Regression tree classification seems to perform well as shown through experiments.

Keywords:

Web Link Spam, Classification, Reptree, Decision Tree, Search Engine

1. INTRODUCTION

Web users rely on the search engines to seek the information from web. In this paper four different regression trees are created, from which it is possible to construct the representative tree which more precisely identifies the spam.

2. PROBLEM DESCRIPTION

2.1 SPAMDEXING

Spamdexing involves a number of methods, such as repeating unrelated phrases, to manipulate the relevance or prominence of resources indexed by a search engine. Search engines use a variety of algorithms to determine relevancy ranking. Some of these include determining whether the search term appears in the META keywords tag, others whether the search term appears in the body text or URL of a web page. Many search engines check for instances of spamdexing and will remove suspect pages from their indexes. Also, people working for a search-engine organization can quickly block the results-listing from entire websites that use spamdexing, vigilant by user complaints of false matches. This paper applies Reptree classification for identifying web spam.

2.2 PAGERANK AND HITS

Two independent efforts in the late 1990 that have profound influence on link based ranking were Brin & Page's PageRank [1] and Jon Kleinberg's work on HITS. PageRank and HITS are the two most important ranking approaches in web search. PageRank was used in Google and HITS was extended and applied in AskJeeves. Modern search engines use not just a single ranking algorithm but a combination of many algorithms and moreover it is not revealed. The simple notation of PageRank is Eq.(1).

$$PR(u) = (1-\alpha) \sum_{v.v \to u} \frac{PR(v)}{O(u)} + \alpha \frac{1}{N} \quad (1)$$

John Kleinberg proposed [2] that web documents had two important properties, called hub and authority. Pages functioning as good hubs have links pointing to many good authority pages, and good authorities are pages to which many good hubs point. Thus, in his Hyperlink- Induced Topic Search (HITS) approach to broad topic information discovery, the score of a hub (authority) depended on the sum of the scores of the connected authorities (hubs):

$$A(u) = \sum_{v:v \to u} H(v) H(v) = \sum_{u:u \to v} A(u). \quad (2)$$

In Eq.(2), $I(v)$ is the in-degree of page v, $O(v)$: out-degree of page v, $A(v)$: authority score of page v, $H(v)$: hub score of page v, W: the set of web pages, N: the number of pages in W, α: the probability of a random jump in the random surfer model, p ! q: there is a hyperlink on page p that points to q. Techniques such as link farms have been developed to subvert both the authority and hub components.

3. RELATED WORK

Gyongyi and Garcia-Molina, illustrated various scenario of the link spam alliances. The methods of the link spam incorporation are addressed. Especially the link farm spam has been considered by them [3]. Yi-Min Wang and Ming Ma proposed a automatic spam detection method Strider Search Ranger. They model the large-scale search spam problem as that of defending against correlated attacks on search rankings across multiple keywords, and propose an autonomic antispam approach based on self-monitoring and self protection. It addresses large-scale spam attacks and redirection spam [4]. Bin Zhou and Zhaohui Tang proposed a spamicity approach for web spam detection. They introduce the notion of spamicity to measure a page is spam. Features are discussed individually and evaluated. They deal with page wise detection strategy [5].

Panagiotis Metaxas uses propagation of distrust to find untrustworthy web neighborhoods. They use backwards propagation of distrust as an approach to finding spamming untrustworthy sites. Their approach is inspired by the social behavior associated with distrust [6]. Jacob Abernethy, Olivier Chapelle and Carlos Castillo proposed graph regularization methods for web spam detection. They propose an algorithm for that named as WITCH. It learns to detect spam hosts or pages on the web. Unlike most other approaches, it simultaneously exploits the structure of the Web graph as well as page contents and features [7]. Jayanthi and Sasikala proposed genetic algorithm based method for link spam detection [8]. They also proposed decision tree induction methods for the link spam classification [8].

4. REPTREE - REGRESSION LOGIC

Regression Trees can be used to model functions, though each end point will result in the same predicted value, a constant for that end point can be achieved. Thus regression trees are like classification trees except that the end point will be a predicted function value rather than a predicted classification. Instead of using the Gini Index the impurity criterion is the sum of squares, so splits which cause the biggest reduction in the sum of squares will be selected. Reptree uses the regression tree logic and creates multiple trees in different iterations. After that it selects best one from all generated trees. That will be considered as the representative. In pruning the tree the measure used is the mean square error on the predictions made by the tree. This paper proposes a method to identify the web spam and since tree based models seems to be more promising; the regression based models are deployed. A tree can be identified with a set of properties which are finite. They have two key components: the tree and its parametric models in each terminal node. The parameters include the splitting rules, topology of the tree (including the children, interior node). The parametric model in each terminal node is the probability of belonging to the each response class.

5. METHODS AND MATERIALS

5.1 REGRESSION TREE CONSIDERATIONS

In regression tree $RT[E;Y]$, E is the leaf of the tree where the tree ends and Y is the response variable. Finding a binary question which gives the maximum information about the Y should be identified and the process should repeat for all levels of the tree. Here Y is considered to be the spam branch of the tree. The leaf E should give the maximum information about this branch that better discriminates the spam and genuine sites. In each children node the process should be repeated in greedy manner. And finally it yields a tree with maximum information gain of spam websites. Since the algorithm is recursive it requires stopping criteria. It is a threshold here. The sum of squared errors for a tree RT is defined as,

$$S = \sum_{E \in leaves(RT)} \sum_{i \in E} (Y_i - N_T)^2 \quad (3)$$

where, N_T is defined as,

$$N_T = \frac{1}{P_c} \sum_{i \in T} Y_i. \quad (4)$$

Eq.(4) is the prediction for leaf N_T. And the modified formula may be like this,

$$S = \sum_{E \in leaves(RT)} p_c V_c \quad (5)$$

where, V_c is considered to be the leaf-within variance and p_c is the class prediction.

5.2 OVERVIEW OF UK-WEBSPAM-2007 DATASET

The detection of the web spam is carried out with the UK-WEBSPAM-2007 dataset [8]. It is based on a set of pages obtained from a crawler of the .uk domain. The set includes 77.9 million pages, corresponding to 11402 hosts, among which over 8000 hosts have been labeled as “spam”, “nonspam” or “borderline”. The link based feature set contains originally 3998 instances with 44 attributes.

Table.1. WEBSPAM-UK-2007 and 2006 dataset comparison

Year	2006	2007
Number of nodes(Hosts)	11,402	114,529
Number of Edges	730,774	1,836,441
Number of labelled Host	10,662	8,479

5.3 PREPARATION OF DATASET

WEBSPAM-UK-2007 dataset is taken as a base. The features used for this work are listed in Table.1. It is preprocessed and converted into five datasets FEATA, FEATB, FEATC, FEATD and FEATE. The specification for the above said features are listed in Table.2. Only link based features are taken for experiments. This paper focus on link spam alone.

Table.2. Feature used from WEBSPAM-UK-2007

Sl. No.	Feature	Type
1	out-links per page	Link
2	intersection of out-links per in-link	Link
3	top-level in-link portion	Link
4	out-links per leaf page	Link
5	in-links per page	Link
6	average level of in-links	Link
7	average level of out-links	Link
8	percentage of pages in most populated level	Link
9	percentage of in-links to most popular level	Link
10	percentage of out-links from most emitting level	Link
11	cross-links per page	Link
12	top-level internal in-links per page on this site	Link

13	average level of page in this site	Link
14	Keyword(s) in title tag	Link
15	Keyword(s) in body section	Link
16	Keyword(s) in H1 tag	Link
17	Keyword(s) in URL file path	Link
18	Keyword(s) in URL domain name	Link

Table.3. Feature set for this work

Sl.No.	Feature Set Name	Focused on
1	$FEAT_A$	Neighbour and Degree Related Features
2	$FEAT_B$	Neighbour and Rank Dependent Features
3	$FEAT_C$	Degree, Neighbour and Rank Dependent Features
4	$FEAT_D$	TPR,PR,TR Features
5	$FEAT_E$	TPR, TR Features

5.4 EVALUATION METRICS

Five different dataset listed in Table.2 are applied for the Classifier. The classifier considered for this work is Reptree (Representative tree with regression logic). Comparison between class detection accuracy was carried out. Evaluation metrics used are listed below,

$$\text{Precision} = \frac{tp}{tp + fp} \tag{6}$$

$$\text{Recall} = \frac{tp}{tp + fn} \tag{7}$$

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn} \tag{8}$$

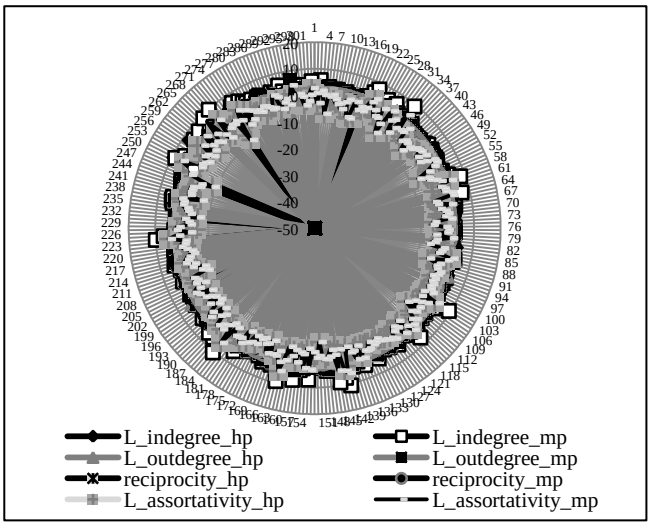

Fig.1. Indegree, outdegree and its correlation with reciprocity and assortativity

The Fig.1 shows the correlation of the degree distribution with the assortativity and reciprocity (same website acting as in link and as well as outlink). Fig.2 shows the indegree distribution of the samples.

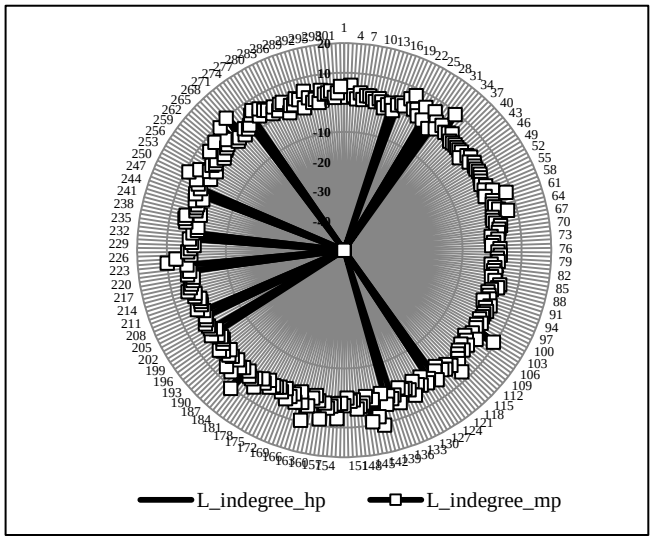

Fig.2. Indegree distribution of spam samples

6. OVERVIEW OF REPRESENTATIVE TREES

Let *RT*1, *RT*2, *RT*3, *RT*4 and *RT*5 be the four trees with *t*1, *t*2, *t*3, *t*4 and *t*5 terminal nodes. They have been trained using the same *n* observations (y_i; x_i); $i = 1 \ldots\ldots n$. For each observation y_i there is an associated fitted value y_{ij} for tree *j*. The trees use the preprocessed datasets FEATA, FEATB, FEATC, FEATD and FEATE respectively. The fitted value is a class label which indicates the web spam. For the generated tree the fitted value is the average of all observations in that node. Since there is a categorical response the expected class label for a node would be the class which had the highest sample proportion. The fitted values of the two trees can be used in fitness metric:

$$FM(RT1, RT2, RT3, RT4, RT5) = \frac{1}{N}\sum_{i=1}^{N} m(y_{i1}, y_{i2}, y_{i3}, y_{i4}, y_{i5}) \tag{9}$$

In Eq.(9), *m* is the metric of fitted values. For regression trees with a continuous response the fitness metric is:

$$m(y_1, y_2) = (y_1 - y_2)^2 \tag{10}$$

This is accomplished by recursively carrying out the following steps at each node,

- Fit a model to the training data there
- Cross-tabulate the signs of the residuals with each predictor variable to find the one with the most significant chi-square statistic
- Search for the best split on the selected variable, using the appropriate loss function
- After a large tree is constructed, it is pruned.

To achieve the maximum information gain the Kullback – Leibler (*KL*) divergence is applied. It is a non-symmetric measure of the difference between two probability distributions

P and *Q*. *KL* measures the expected number of extra fields required to code samples from *P* when using a code based on *Q*, rather than using a code based on *P*. Typically *P* represents the true distribution of data observations. The measure *Q* typically represents approximation of *P*.

For distributions *P* and *Q* of a continuous random variable, KL-divergence is defined to be the integral,

$$D_{KL}(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \ln \frac{p(x)}{q(x)} dx \quad (11)$$

In Eq.(11), *p* and *q* denote the densities of *P* and *Q*. The self-information,

$$I(m) = D_{KL}(\delta_{im} \parallel \{p_i\}) \quad (12)$$

Eq.(12) is the *KL* divergence of the probability distribution *P*(*i*) from a Kronecker delta representing certainty. The mutual information,

$$\begin{aligned} I(X;Y) &= D_{KL}(P(X,Y) \parallel P(X)P(Y)) \\ &= E_x\{D_{KL}(P(Y \mid X) \parallel P(Y))\} \\ &= E_y\{D_{KL}(P(X \mid Y) \parallel P(X))\} \end{aligned} \quad (13)$$

Eq.(13) is the *KL* divergence of the product *P*(*X*) and *P*(*Y*) of the two marginal probability distributions from the joint probability distribution *P*(*X*, *Y*). The conditional entropy,

$$\begin{aligned} H(X \mid Y) &= \log N - D_{KL}(P(X,Y) \parallel P_U(X)P(Y)) \\ &= (i)\log N - D_{KL}(P(X,Y) \parallel P(X)P(Y)) - D_{KL}(P(X) \parallel P_U(X)) \\ &= H(X) - I(X;Y) \\ &= (ii)\log N - E_\gamma\{D_{KL}(P(X \mid Y) \parallel P_U(X))\} \end{aligned} \quad (14)$$

Eq.(14) represents the number of fields which would have to be transmitted to identify *X* from *N* equal likely possibilities. The cross entropy between two probability distributions measures the average number of bits needed to identify an event from a set of possibilities, if a coding scheme is used based on a given probability distribution *q*, rather than the true distribution *p*. The cross entropy for two distributions *p* and *q* over the same probability space is thus defined as follows,

$$H(p,q) = E_p[-\log q] = H(p) + D_{KL}(p \parallel q) \quad (15)$$

7. REPTREE ALGORITHM FOR SPAMDEXING

The regression tree algorithm for web spam detection is as follows.

- Begin with a single tuple containing all link based features of the website. Calculate *NT* and *S*.
- Check the assessment score of the websites and if they are > 0.5 then classify as a spam.
- Otherwise search over all binary splits of all variables for the one which will reduce *S* as much as possible. If the largest decrease in *S* would be less than some threshold α, or one of the resulting nodes would contain less than *q* points, stop. Otherwise, take that split, creating two new nodes. In each new node, go back to step 1.

The paper uses the idea of cross validation from last saved tree. Data is divided into a training set and a testing set (say, 60% training and 40% testing). After that the basic tree-growing algorithm is applied to the training data only, with $q = 1$ and $\alpha = 0$, it grow the largest tree that is possible. This lead to over fit the data. Cross-validation is applied to prune the tree. At each pair of leaf nodes with a common parent, evaluate the error on the testing data, and monitor whether the testing sum of squares would shrink if those two nodes are removed and made their parent a leaf. If so, prune; if not, don't prune. This is repeated until pruning no longer improves the error on the testing data. The reason this is superior to arbitrary stopping criteria is that it directly checks whether the extra capacity (nodes in the tree) pays for itself by improving generalization error.

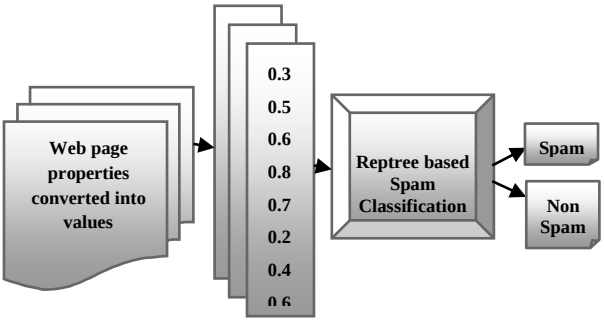

Fig.3. Reptree working method

8. RESULTS AND DISCUSSION

The regression tree results for web spam detection is listed in Fig.4, Fig.5, Fig.6, Fig.7 and Fig.8 for features $FEAT_A$, $FEAT_B$, $FEAT_C$, $FEAT_D$ and $FEAT_E$ respectively.

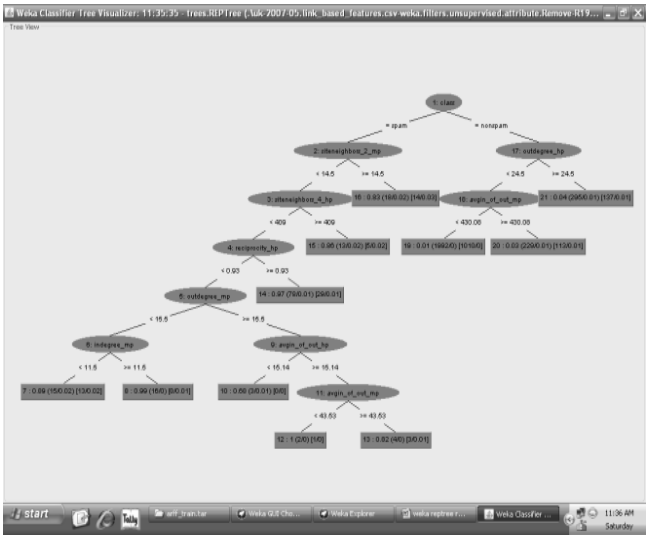

Fig.4. $FEAT_A$ Reptree Result

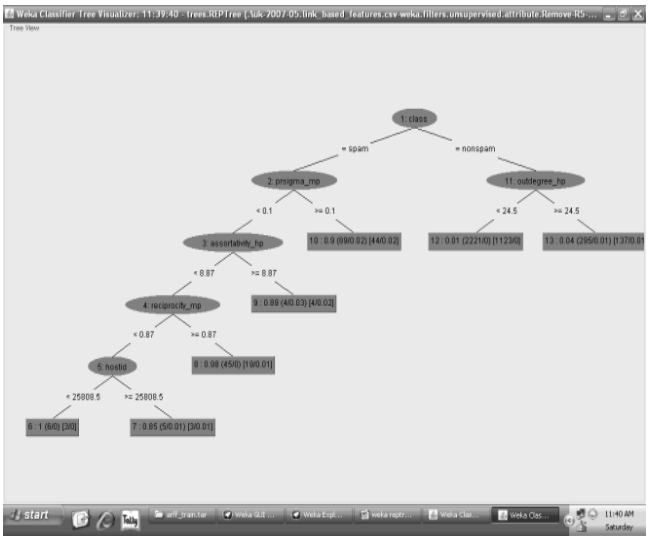

Fig.5. $FEAT_B$ Reptree Result

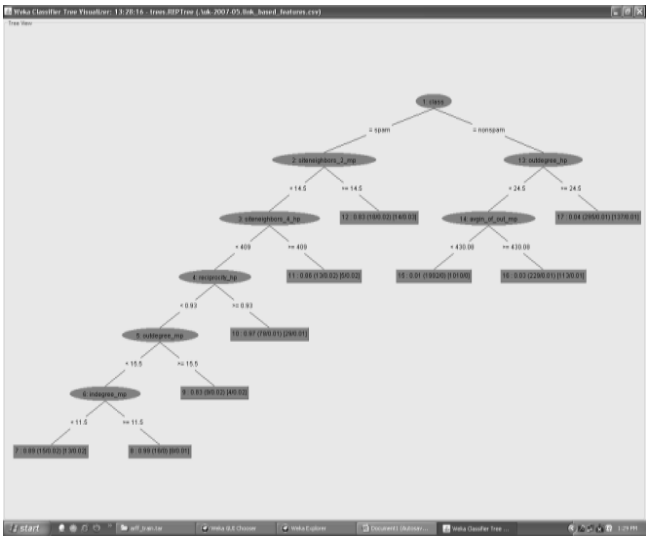

Fig.6. $FEAT_C$ Reptree Result

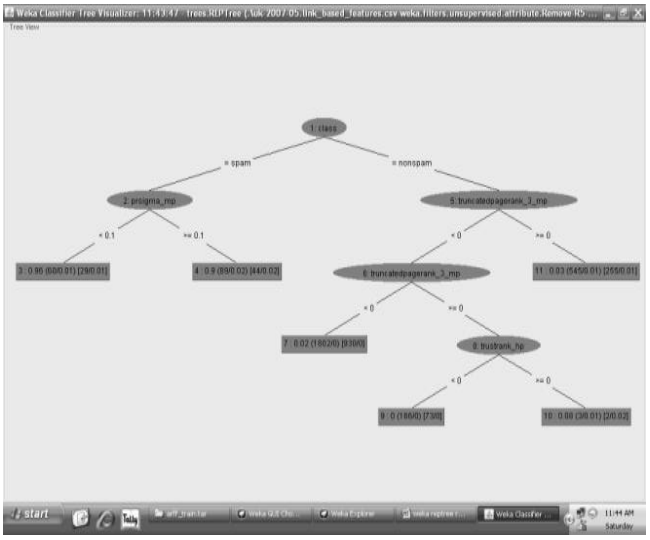

Fig.7. $FEAT_D$ Reptree Result

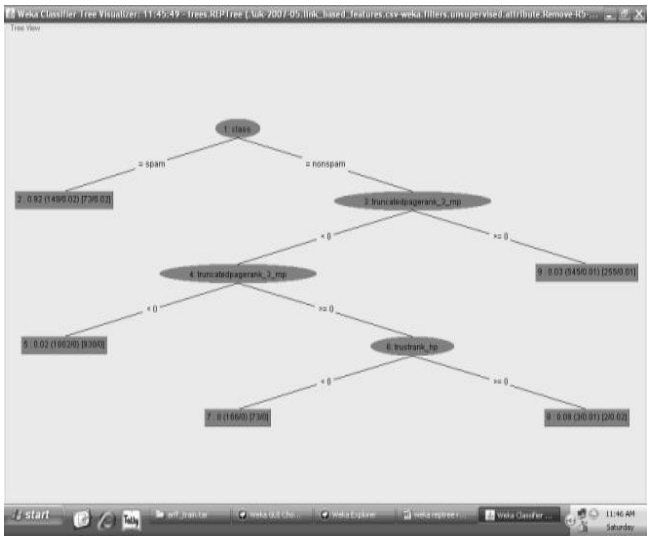

Fig.8. $FEAT_E$ Reptree Result

Results of the Regression Tree – best Reptree

=======

class = spam
| siteneighbors_2_mp < 14.5
| | siteneighbors_4_hp < 409
| | | reciprocity_hp < 0.93
| | | | outdegree_mp < 15.5
| | | | | indegree_mp < 11.5 : 0.89 (15/0.02) [13/0.02]
| | | | | indegree_mp >= 11.5 : 0.99 (16/0) [8/0.01]
| | | | outdegree_mp >= 15.5 : 0.83 (9/0.02) [4/0.02]
| | | reciprocity_hp >= 0.93 : 0.97 (78/0.01) [29/0.01]
| | siteneighbors_4_hp >= 409 : 0.86 (13/0.02) [5/0.02]
| siteneighbors_2_mp >= 14.5 : 0.83 (18/0.02) [14/0.03]
class = nonspam
| outdegree_hp < 24.5
| | avgin_of_out_mp < 430.08 : 0.01 (1992/0) [1010/0]
| | avgin_of_out_mp >= 430.08 : 0.03 (229/0.01) [113/0.01]
| outdegree_hp >= 24.5 : 0.04 (295/0.01) [137/0.01]
Size of the tree : 17
Time taken to build model: 0.14 seconds
=== Cross-validation ===
=== Summary ===

Correlation coefficient	0.9492
Mean absolute error	0.0355
Root mean squared error	0.0689
Relative absolute error	30.0085 %
Root relative squared error	31.4644 %
Total Number of Instances	3998

Fig.9 and Fig.10 represents the error rate comparison on five feature set $FEAT_A$, $FEAT_B$, $FEAT_C$, $FEAT_D$ and $FEAT_E$.

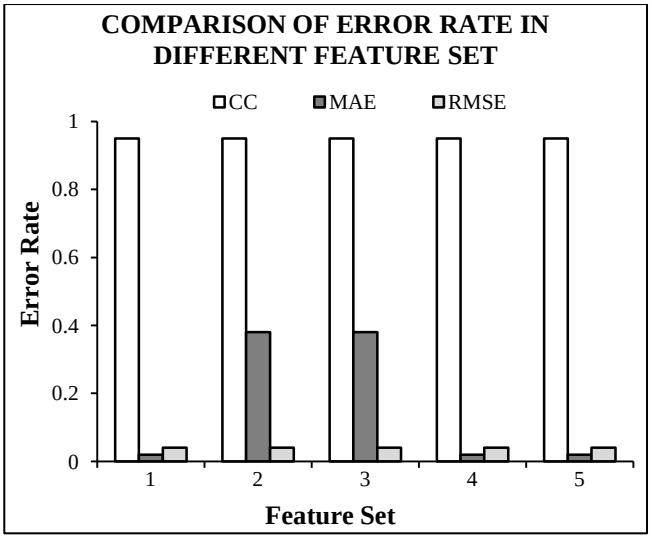

Fig.9. Comparison of Error rate in different feature set

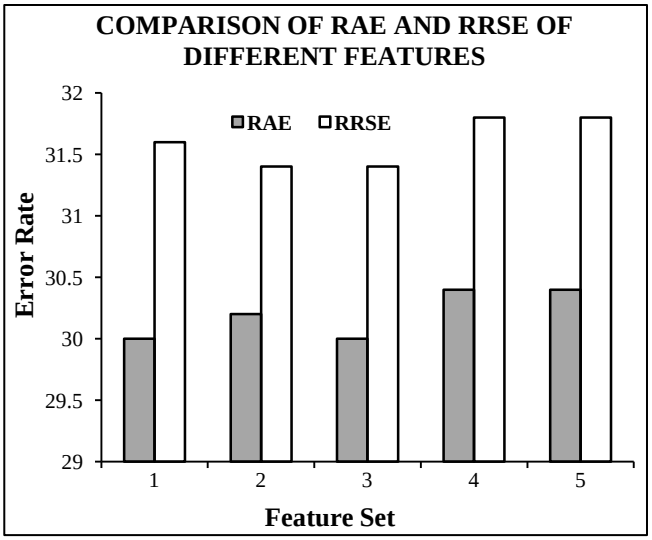

Fig.10. Comparison of RAE and RRSE of different features

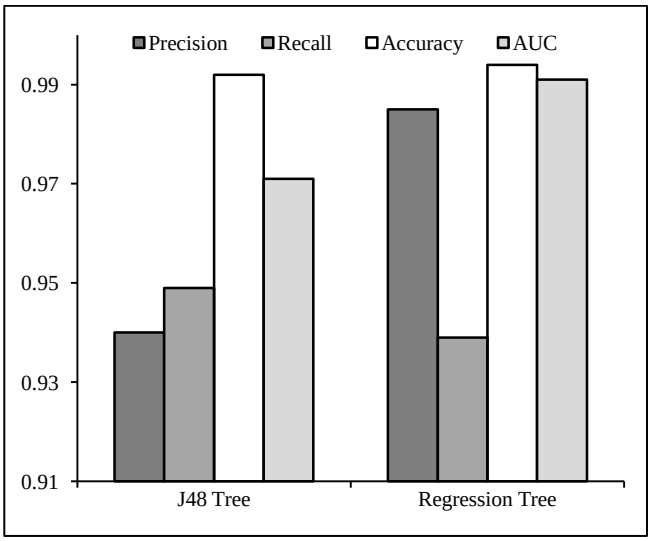

Fig.11. Performance comparison of the J48 and Reptree Results

Fig.11 represents the comparison of the Reptree result (best one) with J48 decision tree classifier. Apart from recall rate the remaining evaluation shows best result for the Reptree.

Table.4. Comparison of error rate in feature sets

Features Considered	CC	MAE	RMSE	RAE	RRSE	TOT_INS
$FEAT_A$	0.9587	0.036	0.0692	30.3724	31.6202	3998
$FEAT_B$	0.9486	0.357	0.0693	30.132	31.4644	3998
$FEAT_C$	0.9492	0.355	0.0689	30.0085	31.4644	3998
$FEAT_D$	0.9482	0.0361	0.0695	30.4847	31.7598	3998
$FEAT_E$	0.9482	0.0361	0.0695	30.4847	31.7598	3998

Table.5. Confusion Matrix

Confusion Matrix			
	a	**b**	**<-- classified as**
a	143	7	a = spam
b	15	135	b = nonspam

Table.6. Performance of Spam/Nonspam Classes

System Performance		
	Precision	**Recall**
a - Spam	95.33%	90.50%
b - Normal	90%	95.07%

Fig.12 shows the precision and recall rate of the Reptree form spam/nonspam classes. Table.4 shows various error rates for the feature sets. The acronyms used in Table.4 are listed below:

- CC-Correlation Coefficient, The correlation is computed between the predicted and actual target values.
- MAE-Mean Absolute Error, it is a quantity used to measure how close forecasts or predictions are to the eventual outcomes.
- RMSE-Root Mean Squared Error, The error is the amount by which the value implied by the estimator differs from the quantity to be estimated.
- RAE-Relative Absolute Error, it is relative to a simple predictor, which is just the average of the actual values.
- RRSE-Root Relative Squared Error, Square root of (Sum of Squares of Errors / Sum of Squares of differences from mean)
- TOT_INS-Total Instance

Based on CC, When all the attributes are used the classification accuracy is higher. For MAE, Rank attributes play a vital role in predicting the spam. In RMSE, RAE and RRSE, when all attributes are included in spam classifier it gives reduced error rate. Table.5 is the confusion matrix of the Reptree and Table.6 shows the precision and recall rate values for spam/nonspam classes.

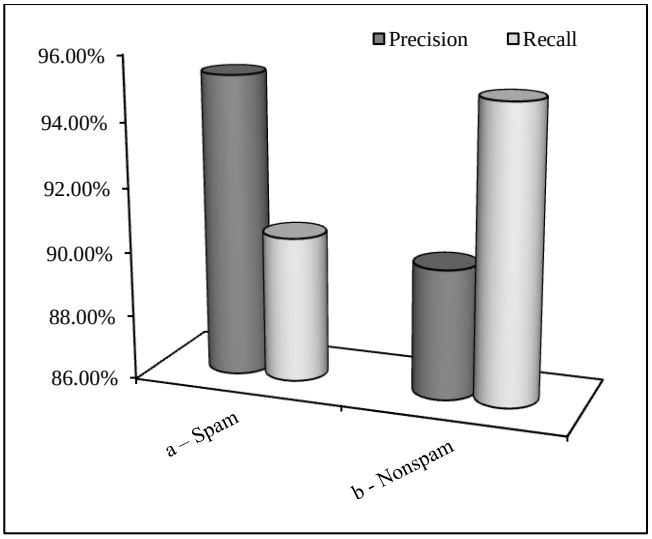

Fig.12. Performance comparison for spam/nonspam classes

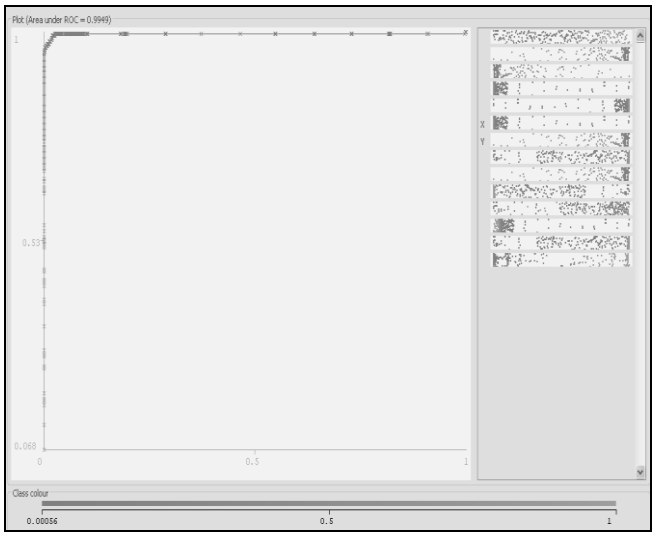

Fig.13. ROC curve

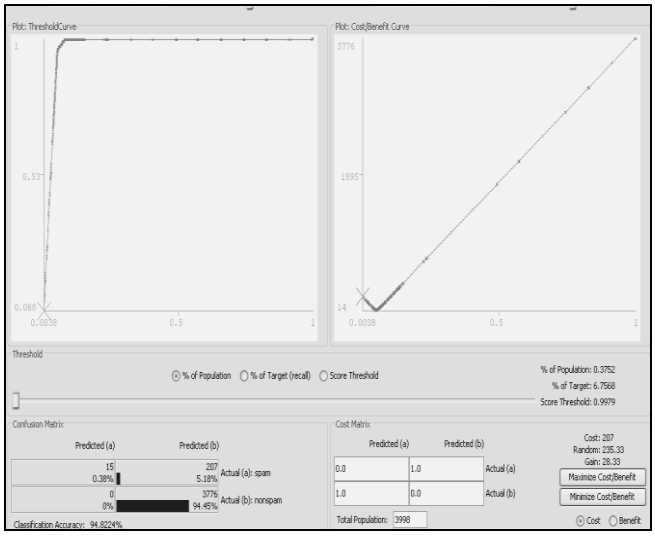

Fig.14. Cost/Benefit Analysis

Fig.13 is the ROC curve for Reptree. AUC value for Reptree is a 0.9949. Fig.14 is the Cost/Benefit analysis graph. Based on that true negative is minimized in Reptree when compared with the J48 decision tree. Fig.15 is the cost curve (probability cost function vs. normalized expected cost) for the spam/nonspam classes. The threshold for this work is 0.5. If a class exceeds 0.5 it is predicted as spam and if it is less than 0.5 then nonspam. If the value is 0.5 then it is considered as the borderline sample.

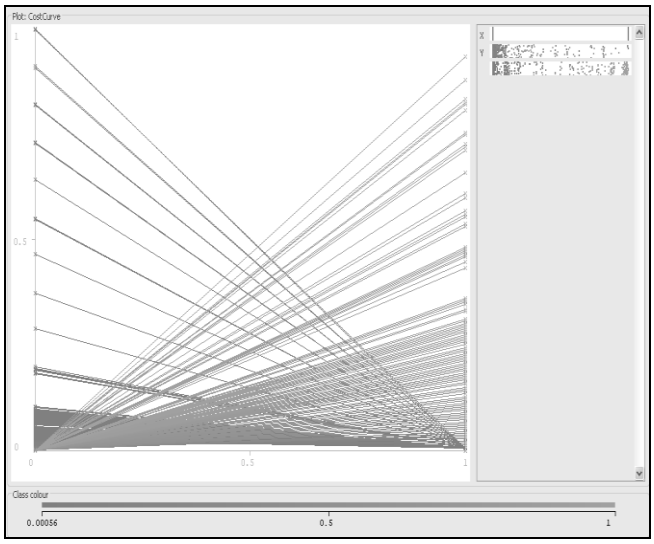

Fig.15. Cost curve for Spam/Nonspam Threshold

9. CONCLUSION

Spamdexing potentially degrades the quality of the results produced by the search engines. This paper addresses a Reptree classification to determine the link spam. In this paper only link based features are considered and hence it cannot detect the content based spam. When both features are combined then it could be possible to achieve more accurate results and this will be the future scope of the paper.

APPENDIX – A

Sample Dataset – After Dimensionality Reduction and PCA

@RELATION .\uk-2007-05.link_based_features.csv
@ATTRIBUTE eq_hp_mp NUMERIC
@ATTRIBUTE assortativity_hp NUMERIC
@ATTRIBUTE avgin_of_out_hp NUMERIC
@ATTRIBUTE avgout_of_in_hp NUMERIC
@ATTRIBUTE indegree_hp NUMERIC
@ATTRIBUTE outdegree_hp NUMERIC
@ATTRIBUTE pagerank_hp NUMERIC
@ATTRIBUTE class {spam,nonspam}
@ATTRIBUTE assessmentscore NUMERIC
@DATA

77,1,0.4375436305999756,0.4375436305999756,12.071428298950195,12.0714
28298950195,64.05555725097656,64.05555725097656,18,18,77,77,
,10242,10242,13,13,1.4255337191536171E-8,1.4255337191536171E-
,0.18871413917322077,1.0,1.0,4,4,17,17,28, ,nonspam,0.000000

112,1,0.6137565970420837,0.6137565970420837,2.200000047683716,2.20000
0047683716,43.875,43.875,24,24,69,69,3040,3040,11134,11134,5,5,3.82915705
7594613E-8,3.829157057594613E-8, , ,spam,1.000000

REFERENCES

[1] Jon M. Kleinberg, "Authoritative Sources in a Hyperlinked Environment", *Journal of the ACM*, Vol. 46, pp. 668-677, 1998.

[2] Page Lawrence and Brin, Sergey and Motwani, Rajeev and Winograd, Terry, "*The PageRank Citation Ranking: Bringing Order to the Web*", Technical Report, Stanford InfoLab, 1999.

[3] Zoltan Gyongyi and Hector Garcia-Molina, "Link Spam Alliances", *Proceedings of the 31st International Conference on Very Large Data Base*, pp. 517-528, 2005.

[4] Yi-Min Wang and Ming Ma, "Strider Search Ranger: Towards an Autonomic Anti-Spam Search Engine", *Proceedings of the 4th International Conference on Autonomic Computing*, pp. 32, 2007.

[5] Bin Zhou, Jian Pei and Zhaohui Tang, "A Spamicity Approach to Web Spam Detection", *Proceedings of the Society of Indian Automobile Manufacturers, International Conference on Data Mining*, pp. 277-288, 2008.

[6] Panagiotis Metaxas, "Using Propagation of Distrust to find Untrustworthy Web Neighborhoods", *Proceedings of the Fourth International Conference on Internet and Web Applications and Services*, pp. 516-521, 2009.

[7] Jacob Abernethy, Olivier Chapelle and Carlos Castillo, "Graph regularization methods for Web spam detection", *Journal on Machine Learning, Springer*, Vol. 81, No. 2, pp. 207–225, 2010.

[8] S.K. Jayanthi and S. Sasikala, "GAB_CLIQDET: - A diagnostics to Web Cancer (Web Link Spam) based on Genetic algorithm", *Proceedings of the 4th International Conference in Global Trends in Information Systems and Software Applications Communications in Computer and Information Science*, Vol. 270, pp. 514-523, 2011.

[9] S.K. Jayanthi and S. Sasikala, "WESPACT: - Detection of Web Spamdexing with Decision Trees in GA Perspective", *Proceedings of the IEEE International Conference on Pattern Recognition, Informatics and Medical Engineering*, pp. 381-386, 2012.

[10] Carlos Castillo, Debora Donato, Luca Becchetti, Paolo Boldi, Stefano Leonardi, Massimo Santini and Sebastiano Vigna, "A Reference collection for Web Spam", *Newsletter ACM SIGIR Forum*, Vol. 40, No. 2, pp. 11-24, 2006.

[11] www.cs.waikato.ac.nz/ml/weka/